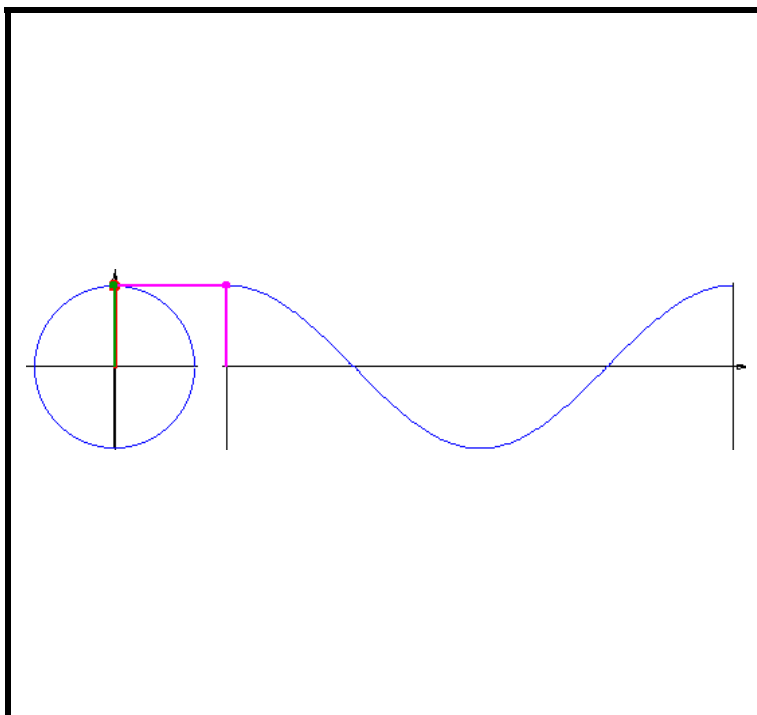


MOTION CONTROL



 früher
EUROTHERM DRIVES

630`SERIES



We thank you for the trust that you have shown in our product.

The operating instructions presented themselves as an overview of the technical data and features.

Please read the operating instructions before putting the product to use.

If you have any questions, please contact your nearest Eurotherm representative.

Improper application of the product in connection with dangerous voltage, can lead to injuries. In addition, damage can also occur to motors or other products.

Therefore please observe our safety precautions strictly.

Topic: **Safety precautions**

We assume that, as an expert, you are familiar with the relevant safety regulations, especially in accordance with VDE 0100, VDE 0113, VDE 0160, EN 50178, the accident prevention regulations of the employers liability insurance company and the DIN regulations and that you can use and apply them. Also the CE - regulations are to be observed and guaranteed.

Depending on the kind of application, additional norms e.g. UL, DIN are to be observed. If our products are employed in connection with components from other manufacturers, their operating instructions are also to be observed strictly.

© **EUROTHERM** Drives Limited.

All rights reserved. No portion of this description may be produced or processed in any form without the consent of the company.

Changes are subject to change without notice.

EUROTHERM has registered in part trademark protection and legal protection of designs. The handing over of the descriptions may not be construed as the transfer of any rights.

Made in Germany, 2003

Further descriptions, that relate to this document

DN: 07-01-08-02



631 - Product manual

DN: 07-01-05-06



635 – Product manual

DN: 07-02-08-03



637 – Product manual

DN: 07-02-09-01



637+ – Product manual

DN: 10-06-03



Product - manual Serial transfer protocol 635 637 637+
EASY-serial

DN: CD



EASYRIDER® Windows – Software

DN:10-06-05



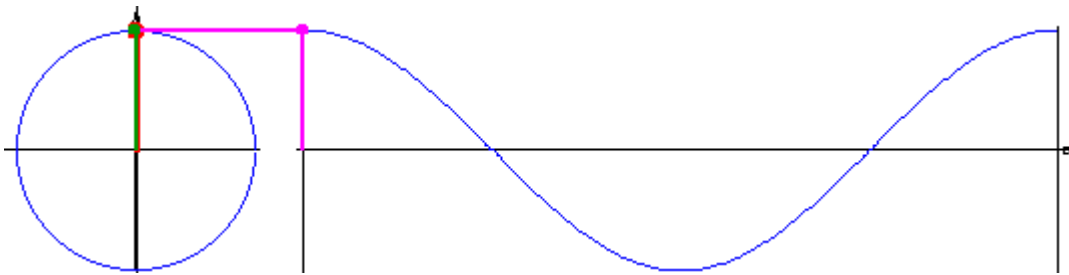
Product - manual Software BIAS®

DN:10-06-06



Product - manual Serial transfer protocol 631

	The most important thing first.....	2
	Further descriptions.....	3
1	Motivation	5
2	Basic control loops and operation mode	6
3	Configuration of the trajectory generator	10
4	Move command block	13
5	Synchronous block	15
6	Memorystructure of CAM–Profiles- and Synchronous functions	21
7	Synchronous-profiles (EASYRIDER® Windows).....	23
8	BIAS access to Synchronous-profiles.....	33
9	Bus access to synchronous-profiles.....	34
10	EXCEL im-export of synchronous-profiles	36
11	Control loop parameters	39
12	Notes	41
13	Modification Record	42



This documentation describes the utilization of the 630 series motion control functions.

All used functions and parameters are based on firmwareversion > V6.15 on all drivetypes of the 630 series and the EASYRIDER® Windows Software Version > V6.15.

Changes are subject to change without notice.

All motion control applications have to be carefully tested. The examples are only a part of the possible combinations, therefore:



CAUTION !

**Programming errors or incompatible operation may cause unpredictable motions.
Avoid danger for operators and machine !**

We assume that, as an expert, you are familiar with the relevant safety regulations of your application. Further the fundamental knowledge of setting up the 630 serie drives with EASYRIDER® Software and BIAS programming is necessary for comprehension of this document.

For convenient support of intended use, EASYRIDER offers 6 general Operation Modes. Depending on the selection (See EASYRIDER-Menue "Comissioning / General" a different set of functions is activated.

Summary Operating Modes

Operation Mode	Function / Purpose	Setpoint	Remark
0	mixed mode, Speed or Current selectable by digital input	Analog Input	
1	Speed	Analog Input	
2	Current	Analog Input	
3	mixed mode, Speed or Position selectable by digital input	Analog Input/ Trajectory Generator	Position mode will be used like in mode 4
4	Position Loop, selection of target-positions by inputs or serial command	Trajectory Generator	predefined selectable positions, easy to use, low flexibility BIAS-Execution not possible
5	selectable by BIAS-programming. All functions available	selectable by BIAS-Language	General purpose, all functions possible by BIAS-programming language

Summary Data of Implemented Functions

Type of Control-loop	Drive Type 631	Drive Type 635 / 637	Drive Type 637+
Current-loop sample-time	211 μ s	211 μ s	105 μ s
Current-loop, Gain of PI-Controller	see 'enhanced explanations' below		
Speed-loop sample-time	633 μ s	633 μ s	422 μ s
Speed-loop, Gain of PI-Controller	see 'enhanced explanations' below		
Position-loop sample-time	1890 μ s	1890 μ s	844 μ s
POSITION-LOOP TIME-CONSTANT OF INTEGRATION	1800 ms	1800 ms	844 ms
Resolution of analog Input	12 bit	12 bit	14 bit
Resolution of Speed-Setpoint	0.5 rpm per Digit	1.44 rpm per Digit	0.56 rpm per Digit

Enhanced explanations

Operation mode 2

The control of the 630 Series is based on a field orientated current control loop. (Figure 2). When the operation mode 2 "Current Control with analog setpoint" is selected, the setpoint is the analog input 1 or if the EASYRIDER® Windows - Software Current Loop menu is started, the operation mode is automatically swapped to "current control" the setpoint is the the output of the function generator in the EASYRIDER® Windows - Software tuning menu. The Gain of the PI-Controller is

$$G=P_Gain*0,245*U_{cc}/I_{NR} *V/A$$

Operation mode 1 "Speed control with analog setpoint" places an other PI controller before the input of the Current Loop. The speed loop is closed through the speed feedback signal, which is derived from the filtered differentiation of the position feedback (resolver, HIPERFACE®, SIN/COS) The filter is implemented as average from 6 to 32 samples adjustable on the 631/ 635 / 637. On the 637+ the filter is implemented fix with 4 samples. The sample time is equal to the current and speed loop sample time, on the 631 / 635 / 637 211µs and on the 637+ it is 105µs.

NOTE:

The analog input 1 is sampled every 633µs on the 631 and 635 /637. On the 637+ every 422µs. The resolution is 12bit on the 631 / 635 / 637 and 14bit on the 637+

The resolution for the speed setpoint on the 631 is 1Digit = 0.5 rpm, on the 635 / 637 it is 1Digit = 1,44 rpm. On the 637+, the value is 1Digit = 0,56rpm. The gain of the PI-speed controller is

$$G=P_Gain*3,35e-4*I_{NR} *A/rpm$$

In **Operation Mode 4 and 5** "Position Control" , "Position Control" with "BIAS Execution" the speed loop is superimposed by a PI Positioncontroller with feedforward. (Figure 1)

The position feedback is given by the increments of the resolver (Option: HIPERFACE®, SIN/COS on 637+).

As a special application it is possible to close the position loop with a external encoder which is placed at the driven shaft and connected to the X40 input. To select this mode "Position control with X40" has to be selected in the EASYRIDER® Windows - Software "counter commissioning".

NOTE:

Setup for this mode has to be done very carefully because positive feedback is possible. Check wiring and count directions of actual position 1 and 2.

The gain of the PI-position controller is $G = P_Gain * 0.045 * rpm / Inc.$ for the 635 / 637 and $G = P_Gain * 0.015 * rpm / Inc.$ for the 631 and 637+. The feedforward is adjusted by the V_Gain parameter. 100% means the feedforward is equal to the speed of the calculated trajectory.

NOTE:

The integration part of the current and the speed loop is adjusted on the EASYRIDER® Windows - Software as the integration time constant in ms.

So the smallest value gives the biggest effect in the control loop.

The position loop is different: Integration Time constant = 1800 ms/I_Gain for 631 / 635 / 637 and Integration Time constant = 844 ms/I_Gain for 637+.

So the smallest value gives the smallest effect in the control loop.

Before the input of the position controller, the “ramp filter” is placed. This filter reduces the jerk in the position setpoint. The filter is implemented as a average of 0-256 samples (Nr. of samples $2^{EASYRIDER_parameter}$).

The sampling time is equal to the position loop sampling time of 1.89 ms for 631/635/637 and 844 µs for 637+.

NOTE:

The new generation of 637+ with 40 MHz processor will have the sampling time for the position loop on 105µs, the sampling time for the rampfilter and trajectory generator will stay on 844 µs.

In Operation mode 4 the trajectory generator can be started by positioning commands of the bus modules and serial interfaces, or by the digital inputs with the positioning blocks, which were set up with the EASYRIDER® Windows - Software .

This mode is created for simple point to point positioning like move absolute or move relative.

For more complex applications the BIAS Interpreter (Bedienoberfläche für intelligente Antriebssteuerungen) = (Operatorsurface for intelligent drive control)

is implemented in the software package of the firmware and EASYRIDER® Windows - Software .

NOTE:

The Operation modes 0 is a mixed mode were analog current or speed control can be chosen by input 24 (635 / 637/ 637+). The Operation mode 3 is a mixed mode were analog speed control or position control (only mode 4) can be chosen by input 24.

This modes have to be handled very careful. Don't forget to tune all the possible control loops of the respective mode.

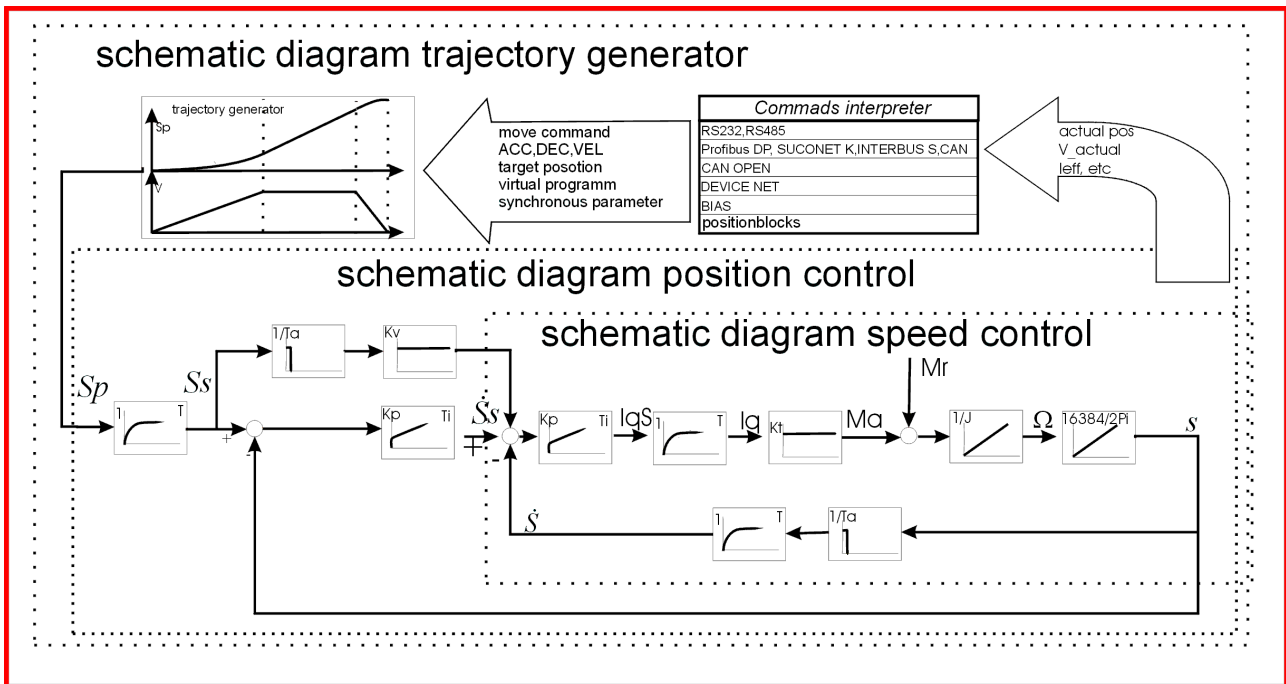


Figure 1

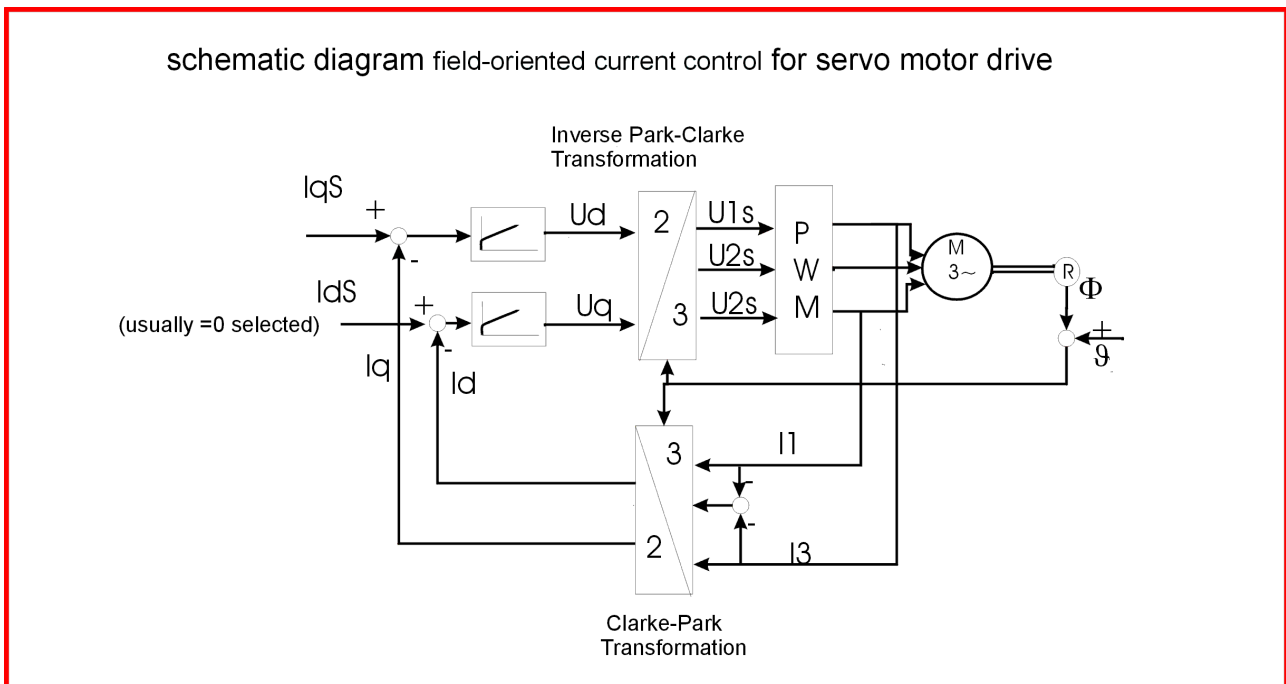
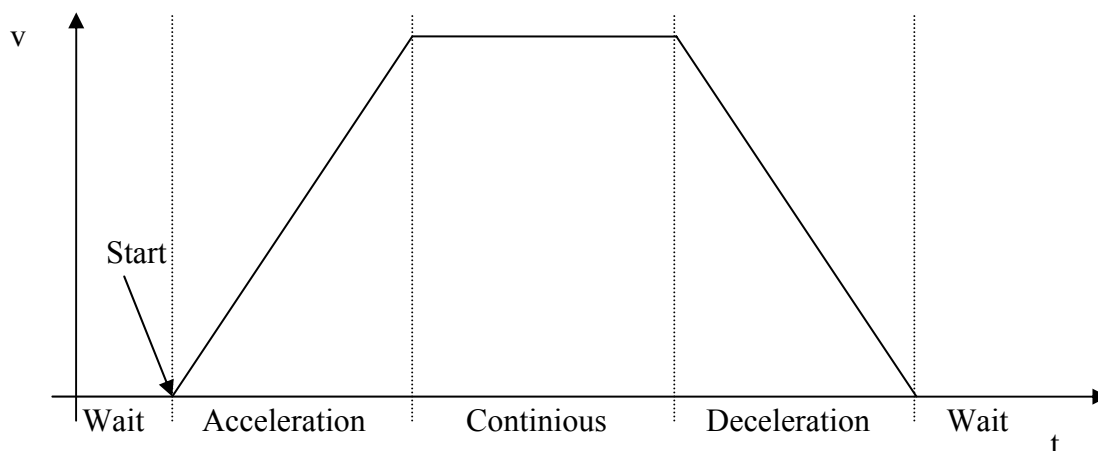


Figure 2

Basic trajectory generator



BIAS command “Virtual Program“

With the BIAS-Interpreter it is possible to start and parametrize Move Commands. The Move Command block is executed in the same software task as the BIAS-Interpreter (1,89 ms 631 / 635 / 637, 0,844ms 637+) and calculates corresponding to the executed Move Command a positionsetpointcurve (trajectory) which the drives follows because of the position control.

A special class of the Move Commands are the Synchronisation Commands. The trajectory is not calculated time based like a “Move absolute“, it is calculated depending a position of another axis.

NOTE:

further in this document we use the mnemonic “positioning command“ for time based movements: Move absolute, Move incremental, Move Datum and “synchronisation“ for Move Synchron and Move Synchronprofil.

On default the synchronisations are part of the Move Commands, that means at one time only a time based positioning command or a synchronisation can be active. With the BIAS-Command „Virtual Program“ it is now possible to have a positioning Command and a synchronisation at one time active.

For example, the output of a “Move relative“ can be connected to the input of a synchronisation. The synchronisation output is connected to position control input. With this configuration it is possible to execute a Cam Profil without an external setpoint source.

NOTE:

switching the X40 output can cause undefined pulses. Therefore the slave axis should not be synchron coupled at that moment.
 general all switching of the command “virtual program“ should be done at standstill of all participated axis.
 switching the X40 output makes only sense, if the X40 is defined as output.
 A synchronisation with actual position 2 as input is not possible in this case.

BIAS-command “Virtual Program = value”

Valid from ab Firmware >= 6.15g for 631/635/637 and >6.13c for 637+

a	b	c	d
Definition of the X40 Output 0 : Standard actual position1 1: Setpoint from positioning 2: setpoint from synchronisation 3: Setpoint from positioning+ setpoint from synchronisation	Positioncontrol input: 0: Setpoint from positioning 1: last setpoint remains unchanged 2: setpoint from synchronisation 3: Setpoint from positioning+ setpoint from synchronisation 4: Setpoint from positioning+ setpoint from synchronisation + Variable 252	Synchroninput: 0: Standard actual position 2 1: Setpoint from positioning 2: act. pos 3	Virtual axis enable: 0: Standard Synchronisation is part of the positioning commands 1: Synchronisation is installed parallel to the positioning commands 3: the virtual synchronisation (d=1) will further be executed after Stop0 or deactivation of the drive.

The command “STOP abrupt” affects axially superimposed. I.e.: The motion of the axis will be stopped with “setpoint = actual value”, the motion control input will be switched-through to default value “0 = Standard” (internal set position), the parallel synchronous block will be switched off. This is not the case, if the option “Further execution of synchronism at stop and deactive” (d = 3) has been selected. In this case the setpoint input (parameter b) is being switched over to the internal set position automatically and the drive is being stopped with “setpoint = actual value”. Nevertheless the synchronous block will be further executed without taking effect on the actual motion set position.

This is also for abrupt stops triggered by the monitoring functions.

It's now possible for example, to limit the controller current and to re-couple the drive (with reduced dynamics) to the master axis (parameter b = 2).

As soon as the system deviation is recompensated, the max. permissible current can be readjusted.

Another possibility is: To read back the actual synchronous position (Variable[X] = Position 3), positioning of the drive on it and afterwards re-switch in the synchronization.

Switching over the motion control input is only possible at standstill ! The standstill of the drive can be checked with the BIAS command “if status 10 = 1 ?”.

If a new synchronous run is being started, it must be switched off with “d = 0” before.

At use of motion control input Mode 4 please note, that Variable 252 = 0 (mandatory!) at switch-on the synchronous run.

At definition of the X40 output “unequal standard actpos1” at 631/635/637, the increment difference of a motion control cycle (1.8 ms) will be pulsated out within 633 µs.

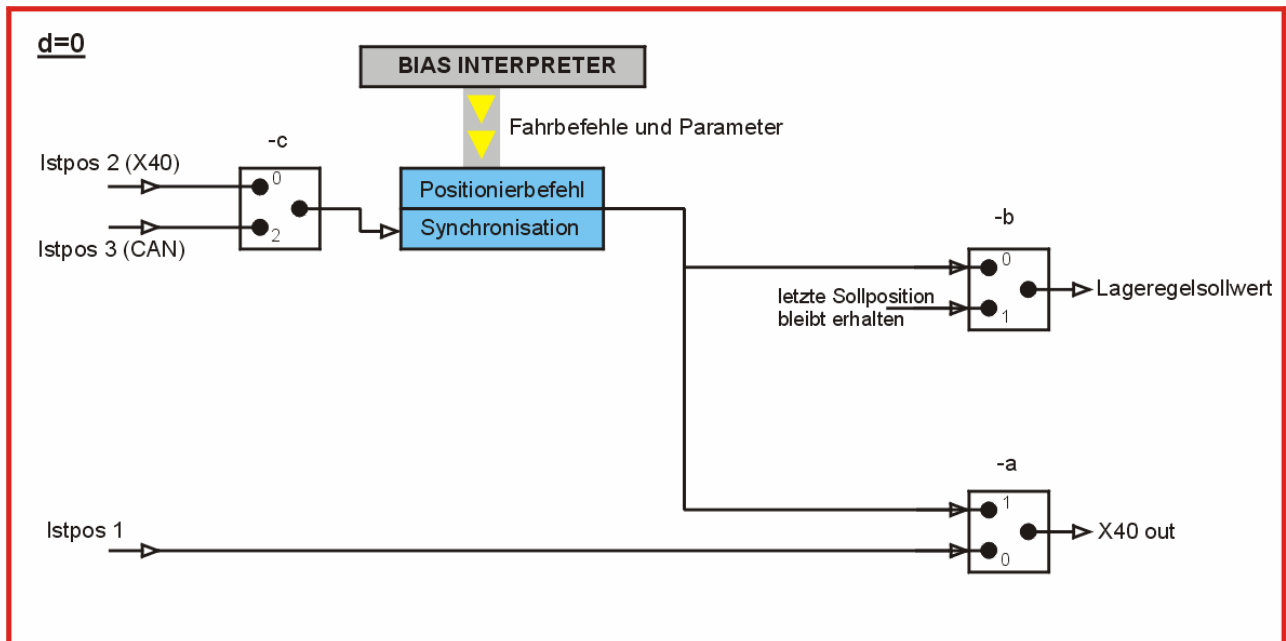
Thereby the permissible limiting frequency of the X40 slave input will be attained 3 times sooner!

This is not valid for 637+ !

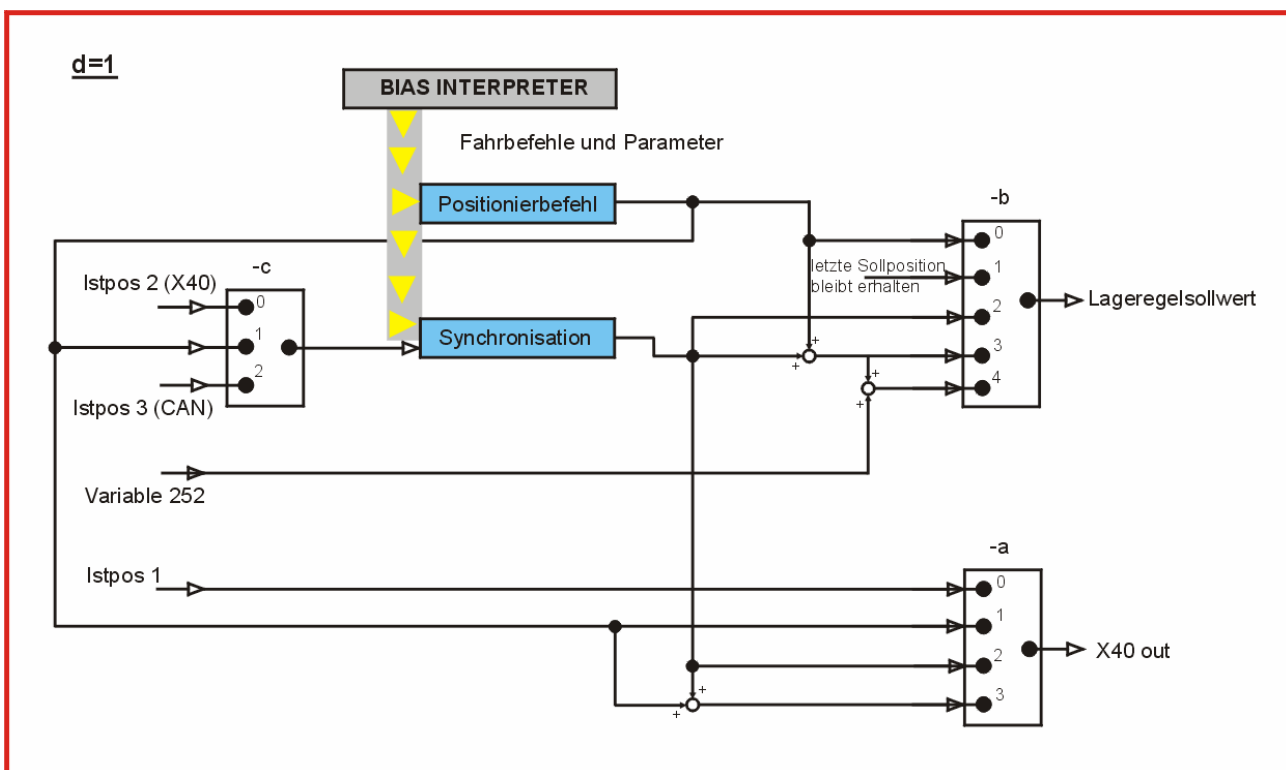
3

Configuration of the trajectory generator

Virtual Program a = x b = x c = x d = 0



Virtual Program a = x b = x c = x d = 1



Hints: if $c = 0$, X40 = input (actual position 2) is **a** out of function.

Positioning command block

The positioning command generates a timebased trajectory, which is calculated out of the parameters position, speed, acceleration and deceleration.

It is implemented as a state machine with the states Wait for Start, Start, Acceleration, Continues and Deceleration.

The BIAS-Interpreter loads the respective parameters and forces the state from “Wait for Start” to “Start”. Now the state machine is stepping itself through the sequence

acceleration→continues→deceleration→Wait for Start. If for example a stop command becomes active during the execution of the positioning command, the BIAS interpreter forces the state simply to deceleration. The axis will stop the state goes back to wait for start.

When the parameters for the positioning are changed during execution, they were overtaken by forcing the state again to start.

The flowchart below shows only a part of the positioning command state machine.

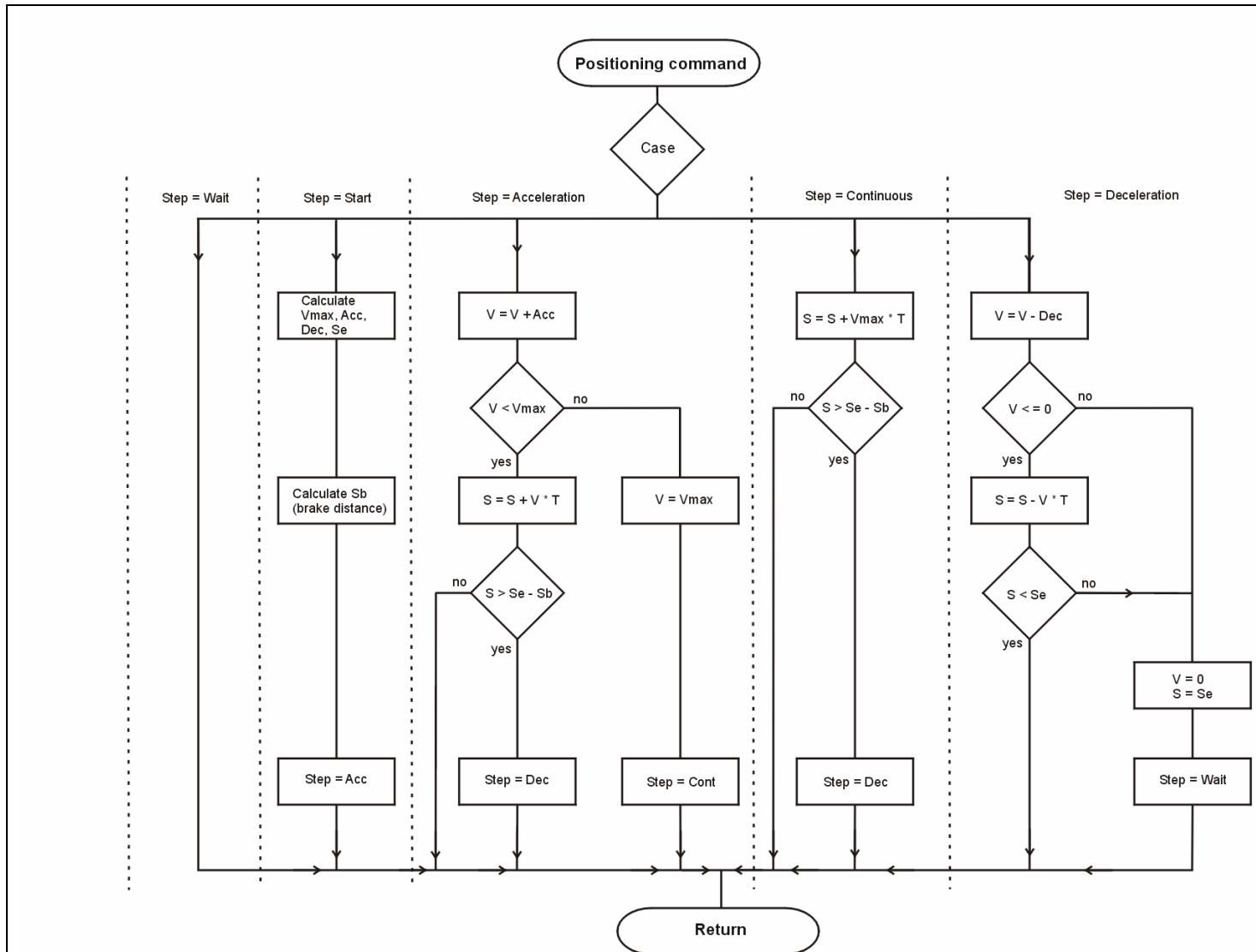
There are more states available for counter clear, PID-Process control, speed or torque controlled moves. Also the synchronisation is on default configuration just a state of the positioning command block.

For Move Datum and remaining positioning separate state machines were implemented which work above the positioning block. They realize the required functionality by automatic concatenation of single positioning and counter initialization.

It is possible to start move absolute, relative, date or move +/- during a synchronisation in default configuration (synchronisation is part of the positioning state machine). The positioning is started with the actual synchronisation speed and accelerates / decelerates to the in the move command defined speed.

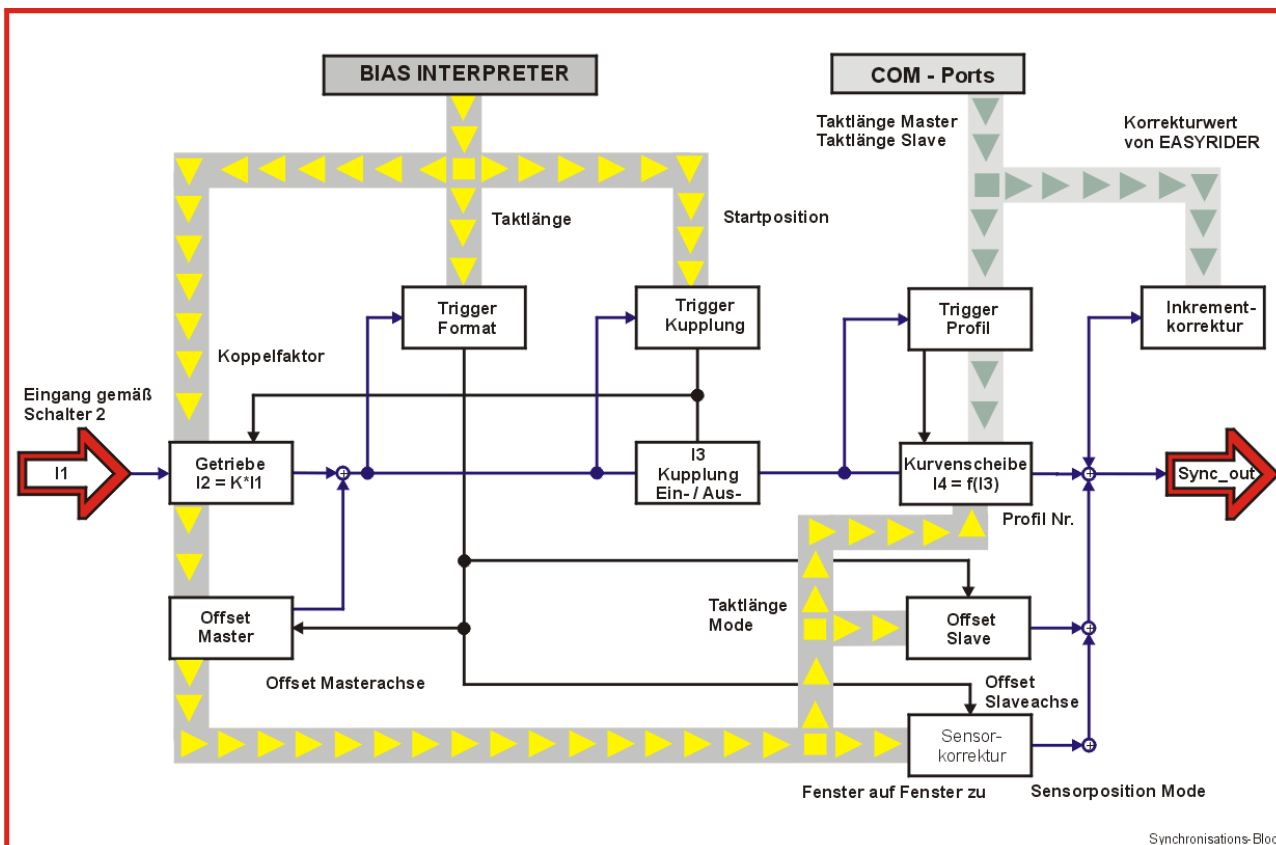
NOTE:

It is important to set the parameter “synchron adjustment 2 mode 11 value =1” for this function. Other wise the position = command will overwrite the internal cycle length of the synchronous move.



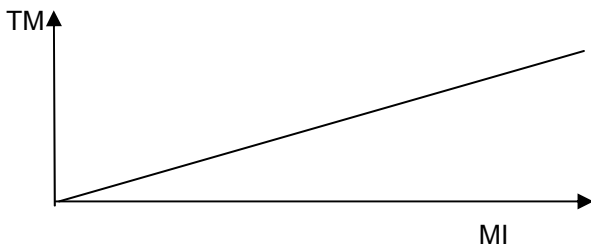
The Synchronous block exists out of a series circuit of the transfer functions gearbox, clutch and cam profile. Additional there are function blocks for the master offset, the slave offset, the sensor correction and the increment correction. They all create position over time functions which can influence the output of the synchronisation through adding blocks.

The comparator blocks Trigger Format, Trigger clutch and Trigger Profile compare the signal flow of series circuit for start- and switching points for starting the time functions of the function blocks or for switching parameters on the exact position.



The Gearbox

The gearbox multiplies the increments coming from the reference source with a fixed factor. This corresponds to a transformation of the master increments (MI) in the Unit of the slave feedback system. The gear ratio can be changed during operation by the BIAS Interpreter or the Bus system. The switch over can be done on a decided position, which is determined through the cycle length parameter, or immediate after executing the BIAS-command „gear ratio=“ in the BIAS-Interpreter. If only the gearbox is activated, the slave axis follows the transferred master pulses(TM). The gear ratio will be adjusted by the commands „gear ratio =“ or „gear ratio=Variable[x] “. With the BIAS command „Synchron. adjust 2, Mode 8 Value=x “the scaling of the gear ratio can be setup. (Details see EASYRIDER help)



$TM = \text{gear factor} * MI$;

The Clutch

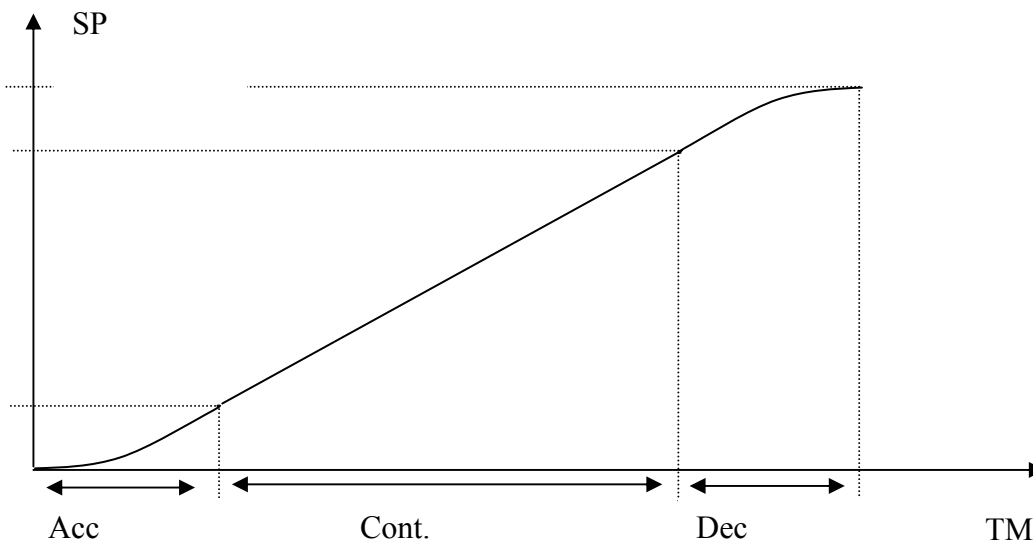
The clutch is used for position- and speed exact up- and down synchronisation to the trajectory of the reference source.

The up synchronisation at the synchronisation start can be done on a by the parameter „start offset“ defined master position if a respective mode is selected in the „synchronadjust „ command. Further it is possible to start with an interrupt input which will be parameterized by the „sensor adjust“command.

With the „start offset“parameters a phase shift between the switch on point of the synchronisation and the start point of the movement can be selected.

The ramp for up- and down synchronisation will be defined by the „cycle length“ – command under consideration of the rule, that as long I3 is moving one cycle length I4 will travel exact half a cycle length with a linear velocity ramp.

The down synchronisation with „synchronadjust mode =1“ does not switch of the synchronisation, the clutch is so to speak, open. Therefore the gearbox and the trigger function are still executed, that allows a position exact re synchronisation.



The CAM-Profile

The CAM-Profile is used to follow user defined master- slave functions, which were executed from a data table.

These tables will be calculated extern and loaded via serial communication to the drive or they will be calculated in the mathematic task of the BIAS-Interpreter.

With the BIAS Command „Move Synchronprofil“ the CAM can be started.

The implemented algorithm is prepared to interpolate linear between the single supporting points of the curve. This guarantees a smooth run of the slave axis.

It is possible to store up to 16 different profiles with a total of 2048 supporting points on the drive.

It can be switched between the single profiles during execution.

Condition therefore is, that this is considered in the profile calculation and that at the switching point a continual jerk free crossover occurs.

The algorithm is generated in that way, that between the single supporting points a linear interpolation is executed. Therefore it is necessary to calculate a respective interpolation factor for each supporting point.

The max. master cycle is limited to 2^{31} /number of supp.points.

Offset Master

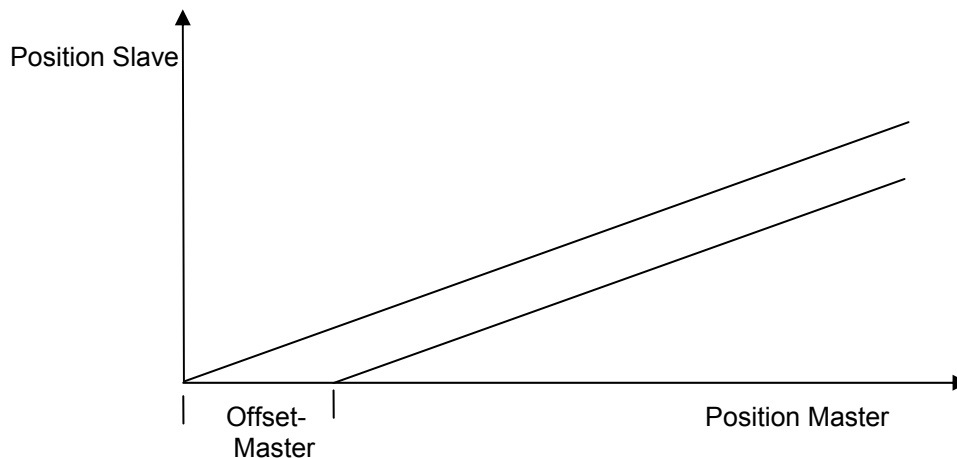
With this time function an offset in direction of the master axis can be adjusted. That means a positioning is overlaid to the synchronisation. The time function is a linear graph between the offset start point and offset end point.

As a default the offset value will be adjusted inside 512 ms. The gradient of the straight line can also be adjusted as a % of the master speed or as a fixed value in increments per second.

The movement of the offset will be started on the format trigger, as long this function block is active.

Thereby the starting point of the time function can be defined exactly.

The BIAS commands “Synchron. Adjust 1 and 2” define the adjustment for the offset function.



Offset Slave

With this time function an offset in direction of the slave axis can be adjusted. That means a positioning is overlaid to the synchronisation. The time function is a linear graph between the offset start point and offset end point.

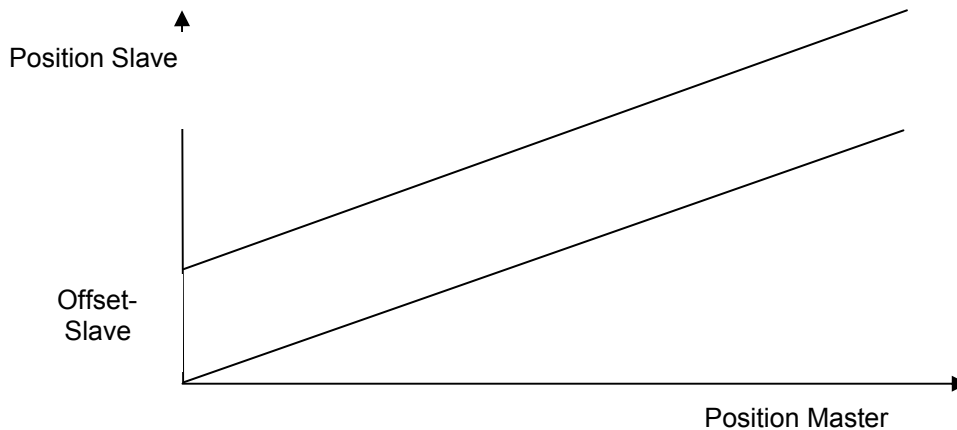
As a default the offset value will be adjusted inside 512 ms. The gradient of the straight line can also be adjusted as a % of the master speed or as a fixed value in increments per second.

To reduce the jerk, which occurs by executing this function, the ramp filter can be aligned with the offset slave function.

The movement of the offset will be started on the format trigger, as long this function block is active.

Thereby the starting point of the time function can be defined exactly.

The BIAS commands "Synchron. Adjust 1 and 2" define the adjustment for the offset function.



Sensor Correction

The sensor correction determines the position error of the driven material or work pieces through the evaluation of the through a photo sensor initiated interrupt at $-X10.25$. The resulting time function is a straight line between start and end point, like the offset functions.

The alignment with the ramp filter is in this case also allowed.

The execution time for a single correction will be adjusted with the BIAS command "Sensor adjust 2" in ms.

Increment Correction (EASYRIDER-function)

With the increment correction it is possible to compensate the position error, which is caused through a gear box with a gear ratio which can not be represented by the "gear ratio" parameter of the electronic gearing function of the synchronisation block.

For this, the EASYRIDER Software calculates multistage correction values and loads them with the CAM-profil to the drive.

This function is nowadays not necessary anymore, because the "gear ratio" parameter is implemented as a fraction and a uneven gearing can be also implemented by a linear profile.

The function is held in the software package for compatibility to older versions.

Trigger-Format

The Trigger-Format compares the output of the gearbox function to the by the “Cycle length” command defined format.

On switching point

- The offset functions for master and slave

- The sensor correction and

- The execution of the calculated corrections

will be activated if the trigger function is selected.

The trigger bit can be evaluated in the BIAS program to control the loading of the parameters of the above mentioned functions.

Trigger-Clutch

The Trigger-Clutch compares the output of the gearbox function for the in the parameter “Startoffset” defined phase shift between the Trigger-Format and start of the clutch function or the reload of the gear ratio.

Here also a pulse signal is generated to control the reload of the parameters in the BIAS program.

Trigger-Profile

The Trigger-Profile compares the output of the clutch to the master stroke of the profile which is loaded together with the parameters of the cam profile.

Will the master stroke be reached the corresponding trigger bit will be set. This trigger can also be used in the BIAS program for switching between different CAM-profiles.

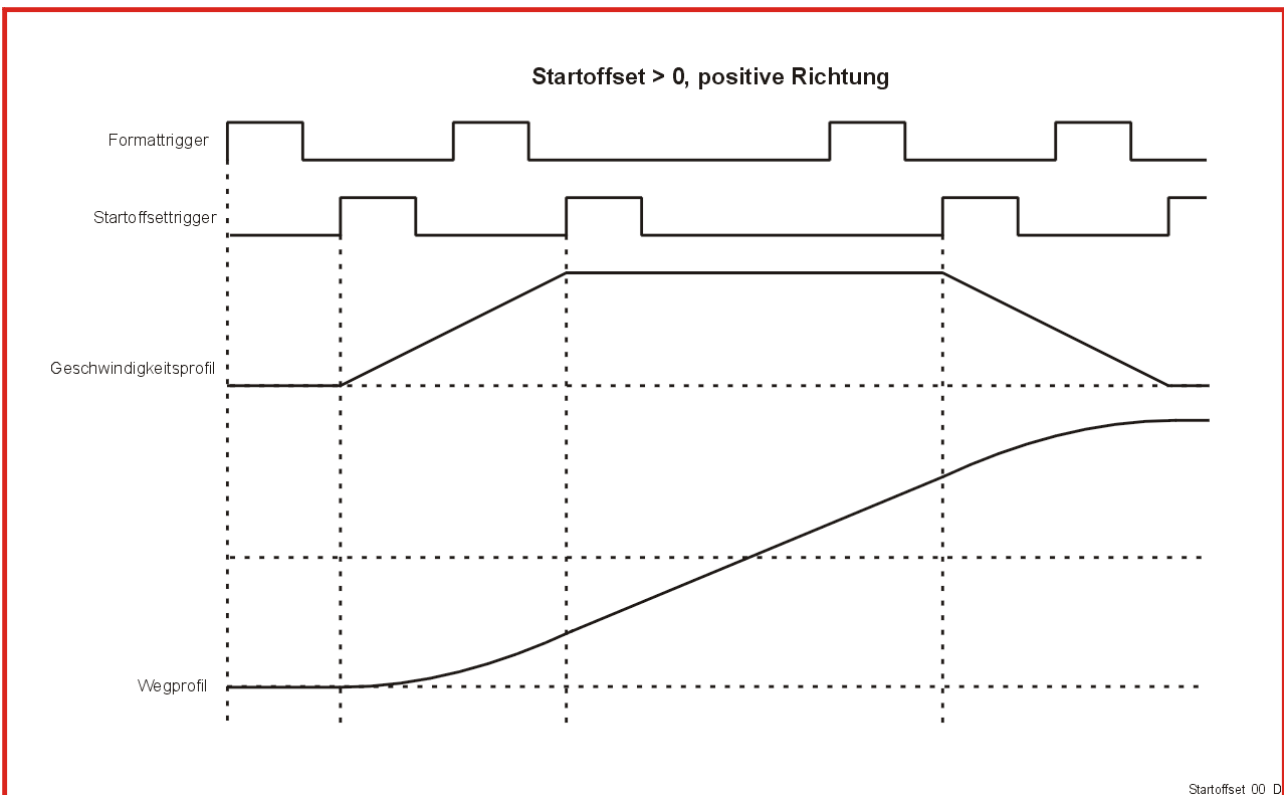
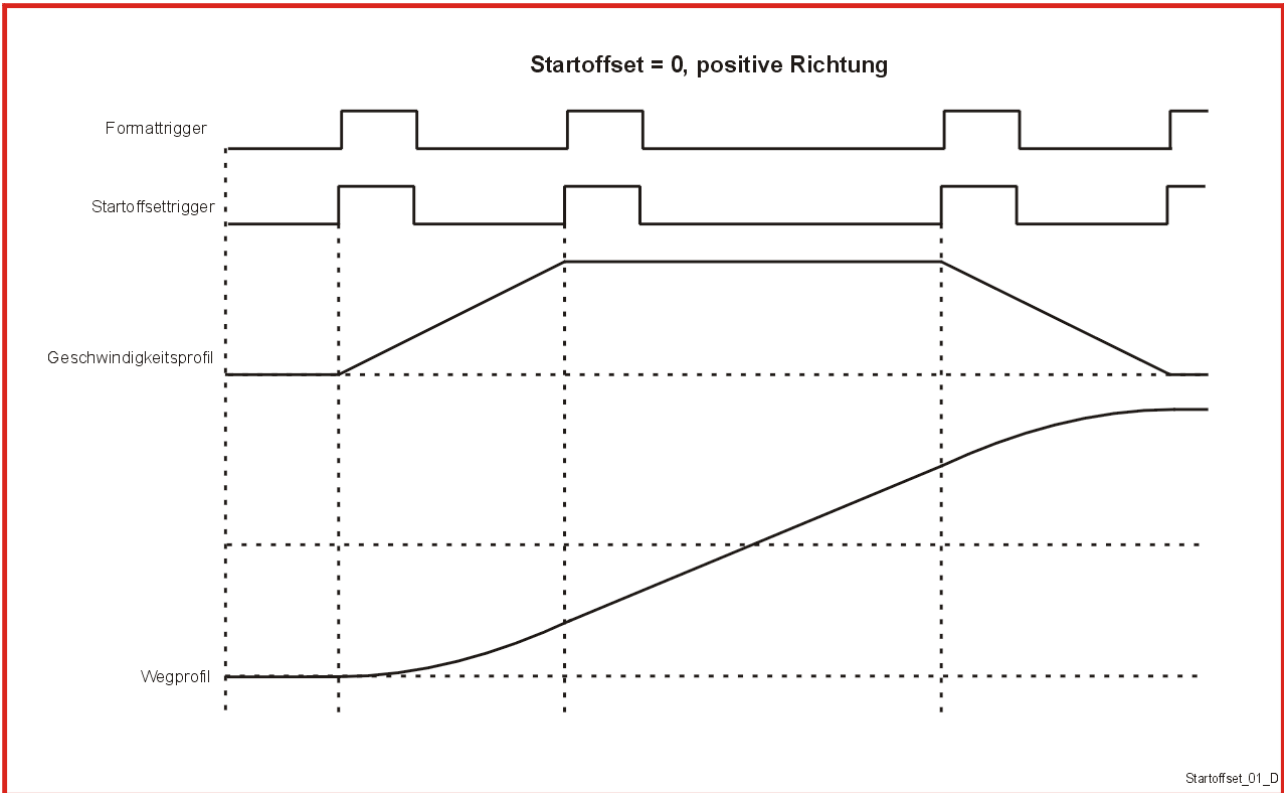
On the trigger point the software proofs if another profile should be executed and if yes, this new profile will be started on the exact position.

After the resumption of the trigger bit a new profile number can be loaded in the BIAS program.

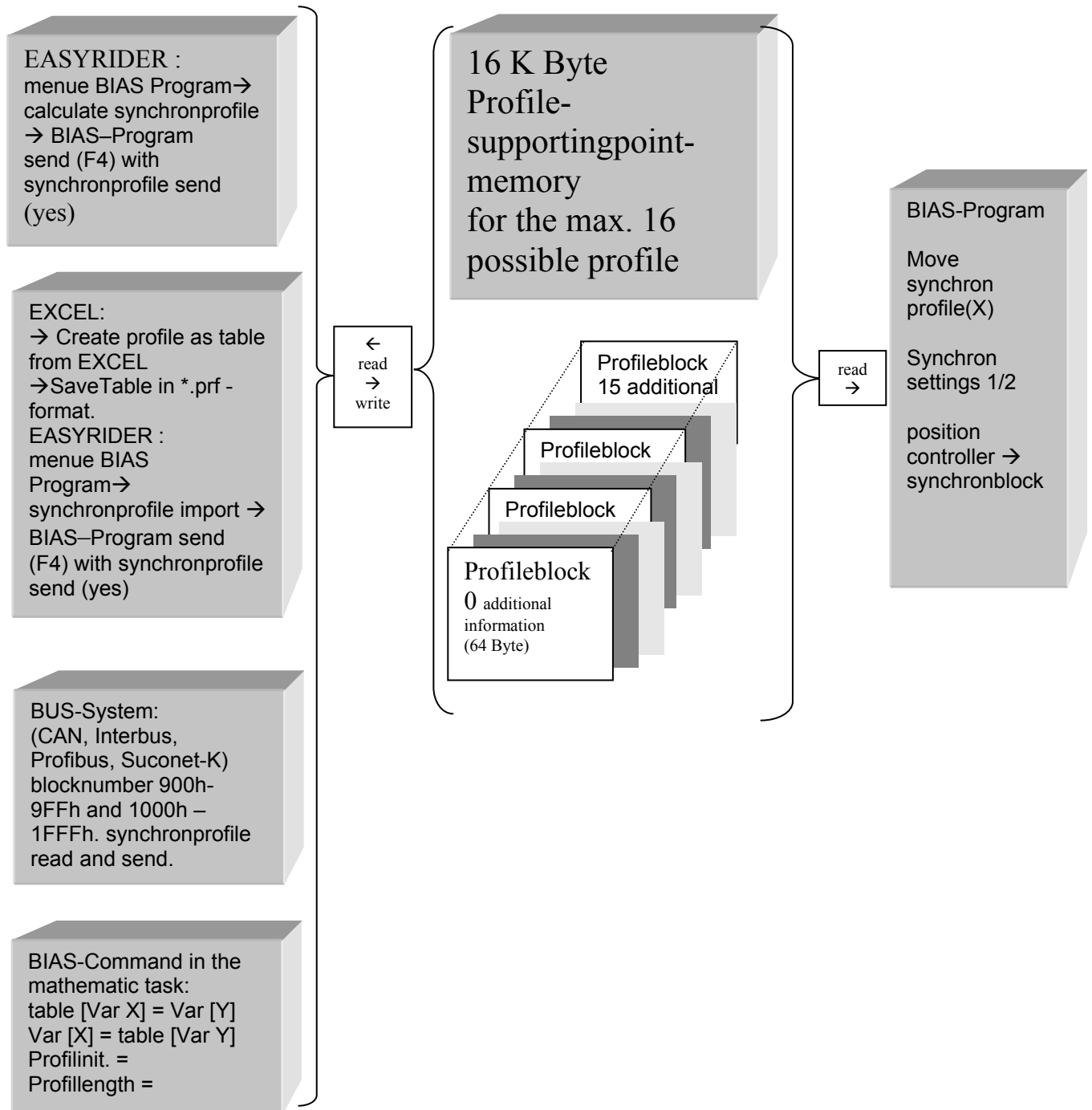
This profile will then be activated on the next Trigger-Profile.

The trigger status of all three trigger functions can be checked by the BIAS commands “If Status x ? then jump” and “flag x = Status y”

NOTE: The Trigger cycles should be chosen bigger then the amount of transferred master pulses for one sampling cycle. Otherwise the triggering will not work correctly.



Access possibilities to the synchronous profile data



-The max. formatlength is 2^{31} /number of profile points.

NOTE: for firmware <6.15h on 631/635/637; <6.13 on 637+

-The max. Interpolationfactor should be less then 2^{23}

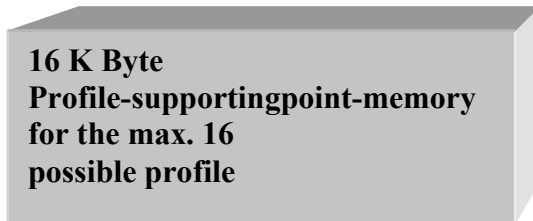
-The value (format length/number of profile points) must be lower than 32768 incr.

-The max. number of profile points = 2x 1024. That means 2 Profiles with max. 1024 points are possible.

Memory structure of the profile area in the devices of the 630 Serie

The memory structure in the devices of the 630 series is splitted in two separat parts:

1. Profilememory for all supportingpoints (memory 16 KB)

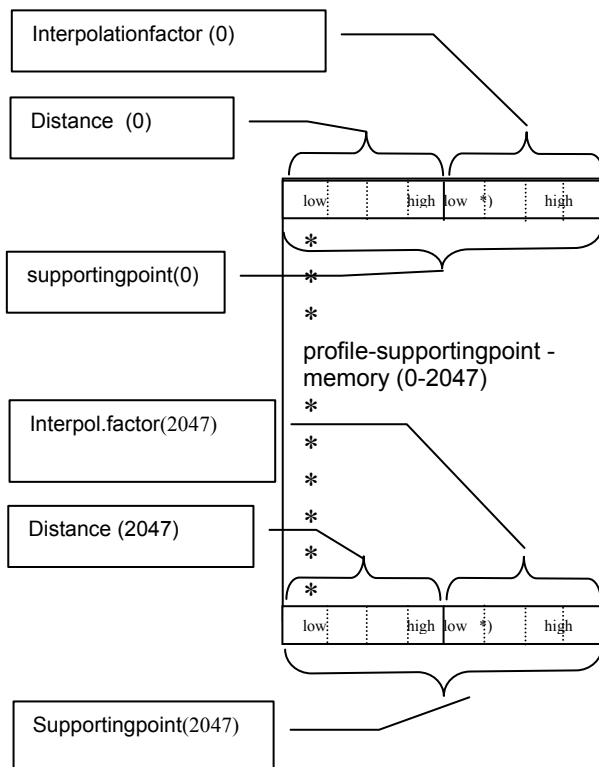


The profile-supportingpoint memory has the following structure:

16 K Byte memory = 16384 Byte / 8 Byte = 2048 supportingpoints

Every supportingpoint is defined with 4 Byte distance (in incr.) and 4 Byte interpolationfactor.

Note: The integer part, of the interpolationfactor is fixed in the 3.rd byte *).



Slaveposition(i) { SP(i) }: Slaveposition of a profilepoint in increments. Dataformat is long integer.

$$SP(i) = f(MP(i));$$

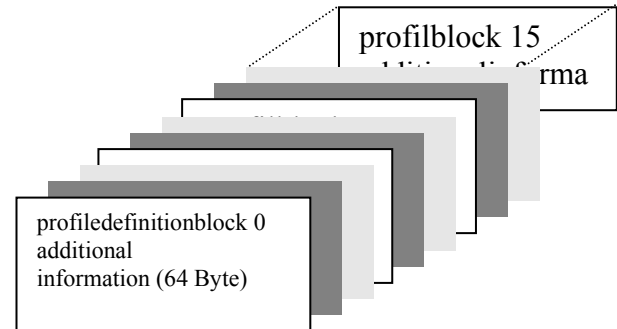
Interpolationfactor(i) { IP(i) }: gradient between two profilepoints*65536. Dataformat long integer.

$$IP(i-1) = 65536 * \{SP(i) - SP(i-1)\} / \{MP(i) - MP(i-1)\};$$

Condition:

$$\{MP(i) - MP(i-1)\} = \text{constant !}$$

2. Profileblockmemory for the additional information of the max. 16 profiles (memory 1 KB)



Paramters of the profiledefinitionblocks:

Profilpoints { PP }: Number of points of the slaveposition-masterposition-graph.

Syncstartaddress { STS }: First point of the profile. Are all profiles of the same length the syncstartaddress=(profile_nr-1)*profilepoints. In the other case syncstartaddress is equal to the sum of profilepoints of the lower profiles.

Masterstroke { MT }: Length of a masterstroke in increments. Corresponding the definition area of the function slaveinc=f(masterinc).

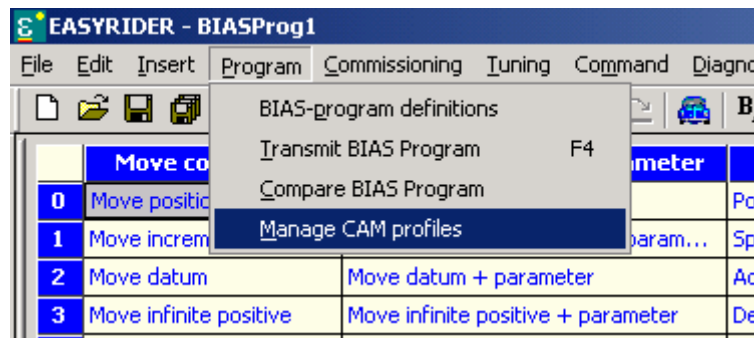
Slavestroke { ST }: Length of a slavestroke in increments.

On strict monotone function corresponding the value area of the function slaveinc=f(masterinc).

Is the function not strict monotone ST is equal to the end point of the profile.

For example a profile were the slave going backwards to his startposition, the slavestroke is zero.

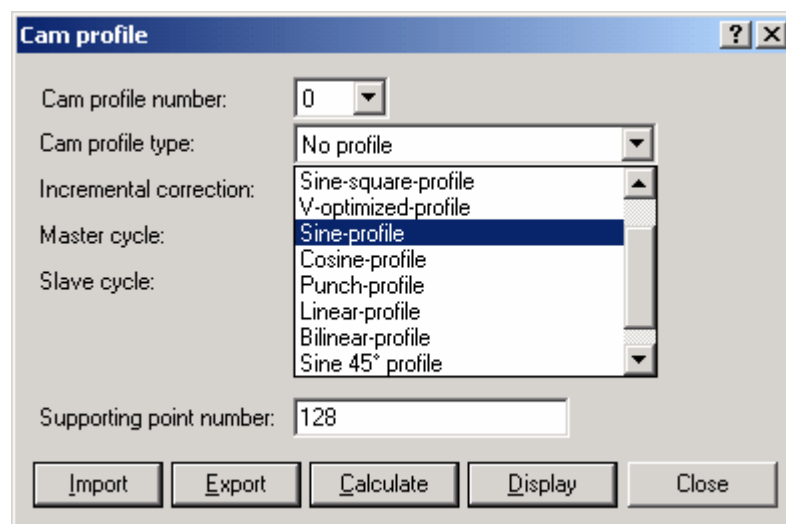
Profiletype { TY }: This Byte defines the type of profile. Is the profile calculated by the EASYRIDER Software the type number of the shape is placed in here. For user defined profiles (calculated from EXCEL or BIAS Math-task) insert the value **255d**.



The **EASYSRIDER® Windows** is able to manage and calculate some predefined profilename. The user just has to select the profilename and must parametrize the expected inputdatas. The calculation is done by the **EASYSRIDER** automatically.

The following list shows the available types of profiles in the **EASYSRIDER® Windows** software.

- PCM profile**
- Sine-square-profile**
- V-optimized-profile**
- Sine- profile**
- Cosine- profile**
- Punch- profile**
- Linear- profile**
- Bilinear- profile**
- Sine 45°- profile**
- Cosine 45°- profile**



PCM profile

Cam profile [?] [X]

Cam profile number: 0

Cam profile type: PCM-profile

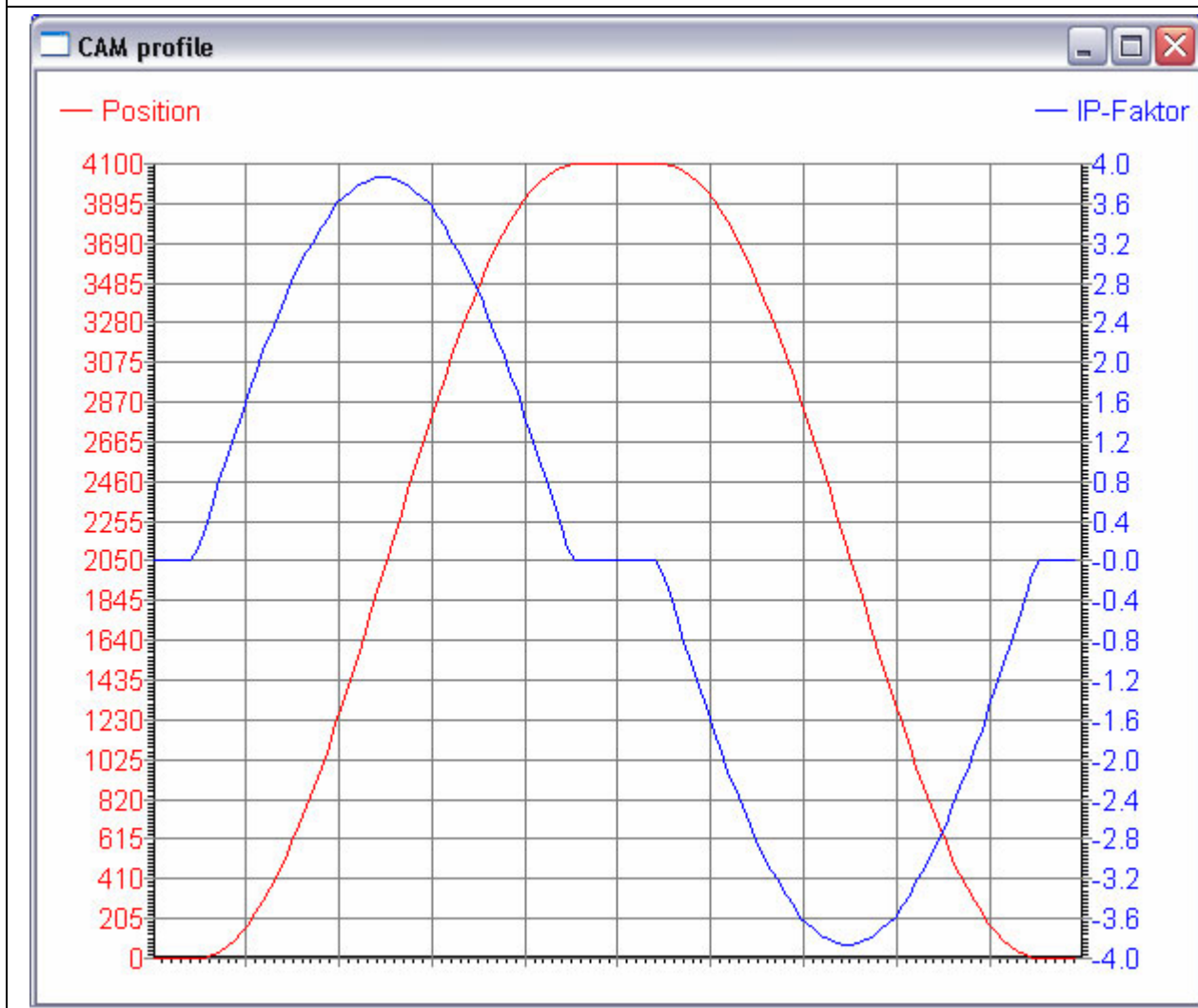
Incremental correction: No

Master cycle: 4096 incr

Slave cycle: 4096 incr

Supporting point number: 128

Import Export Calculate **Display** Close



Sine-square-profile

Cam profile ? X

Cam profile number: ▾

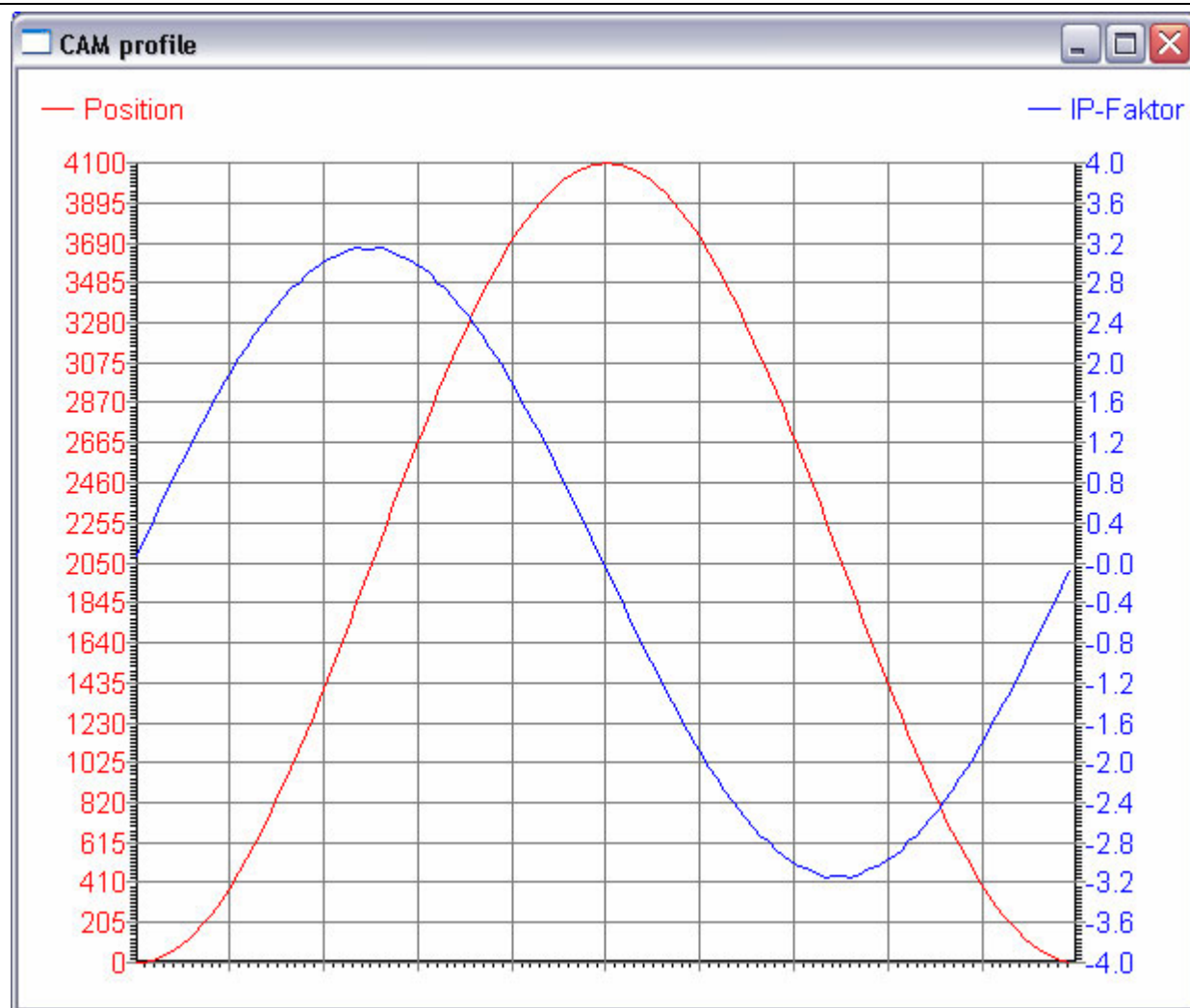
Cam profile type: ▾

Incremental correction: ▾

Master cycle: incr

Slave cycle: incr

Supporting point number:



V-optimized-profile

Cam profile [?] [X]

Cam profile number: 0

Cam profile type: V-optimized-profile

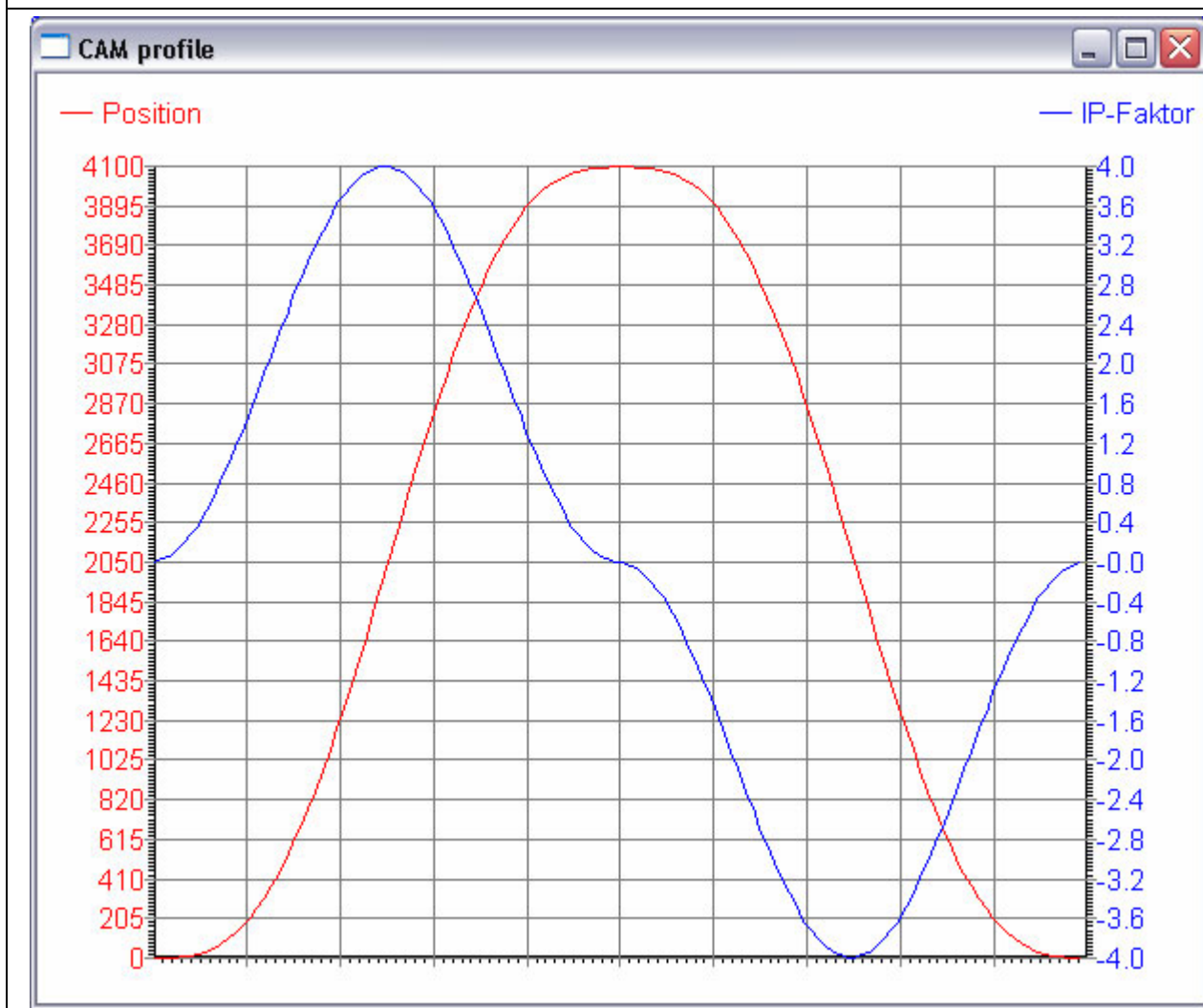
Incremental correction: No

Master cycle: 4096 incr

Slave cycle: 4096 incr

Supporting point number: 128

Import Export Calculate Display Close



Sine-profile

Cam profile ? X

Cam profile number:

Cam profile type:

Incremental correction:

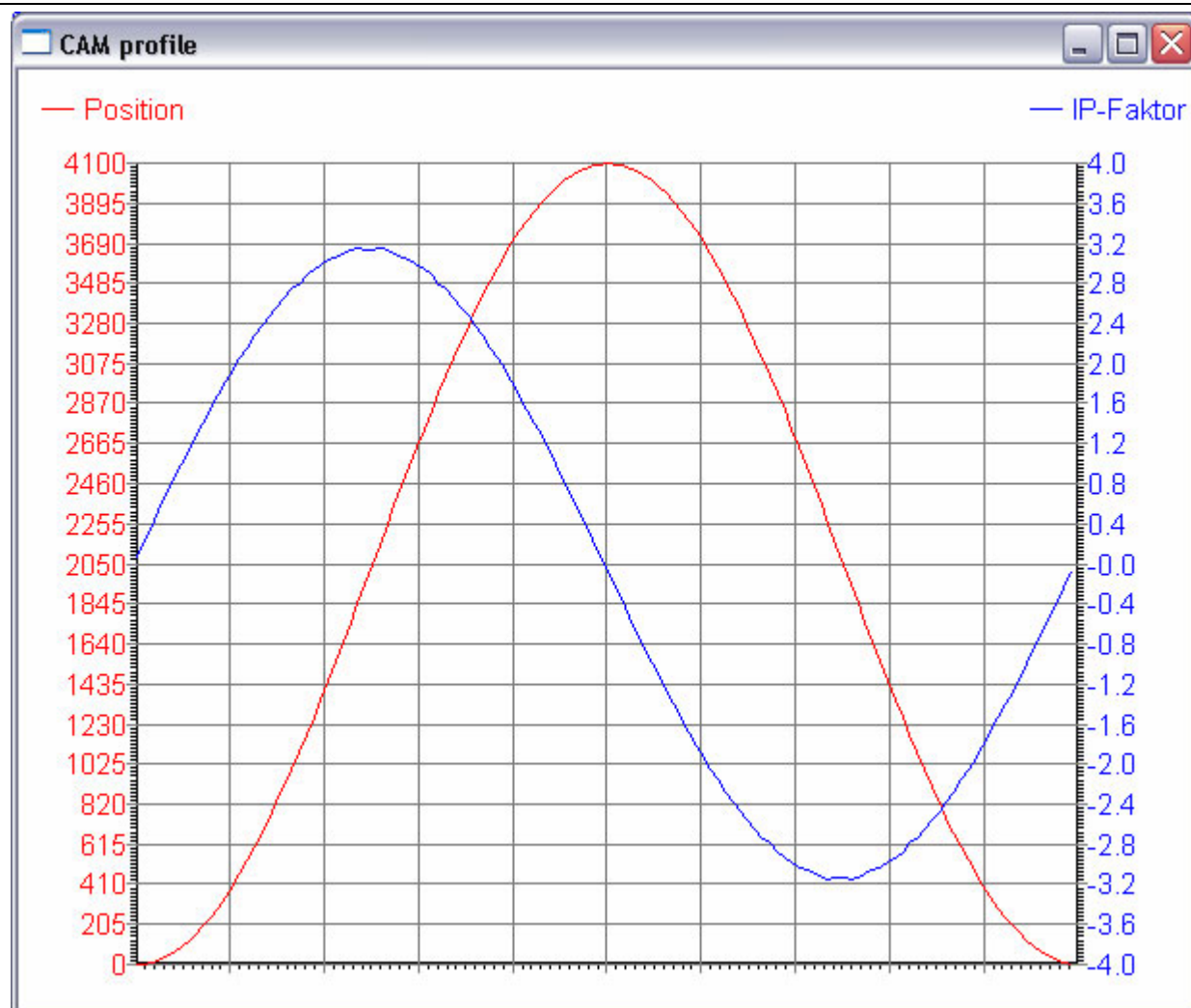
Master cycle: incr

Slave cycle: incr

constant part cosine: incr

sine part: incr

Supporting point number:



Cosine-profile

Cam profile ? X

Cam profile number: ▾

Cam profile type: ▾

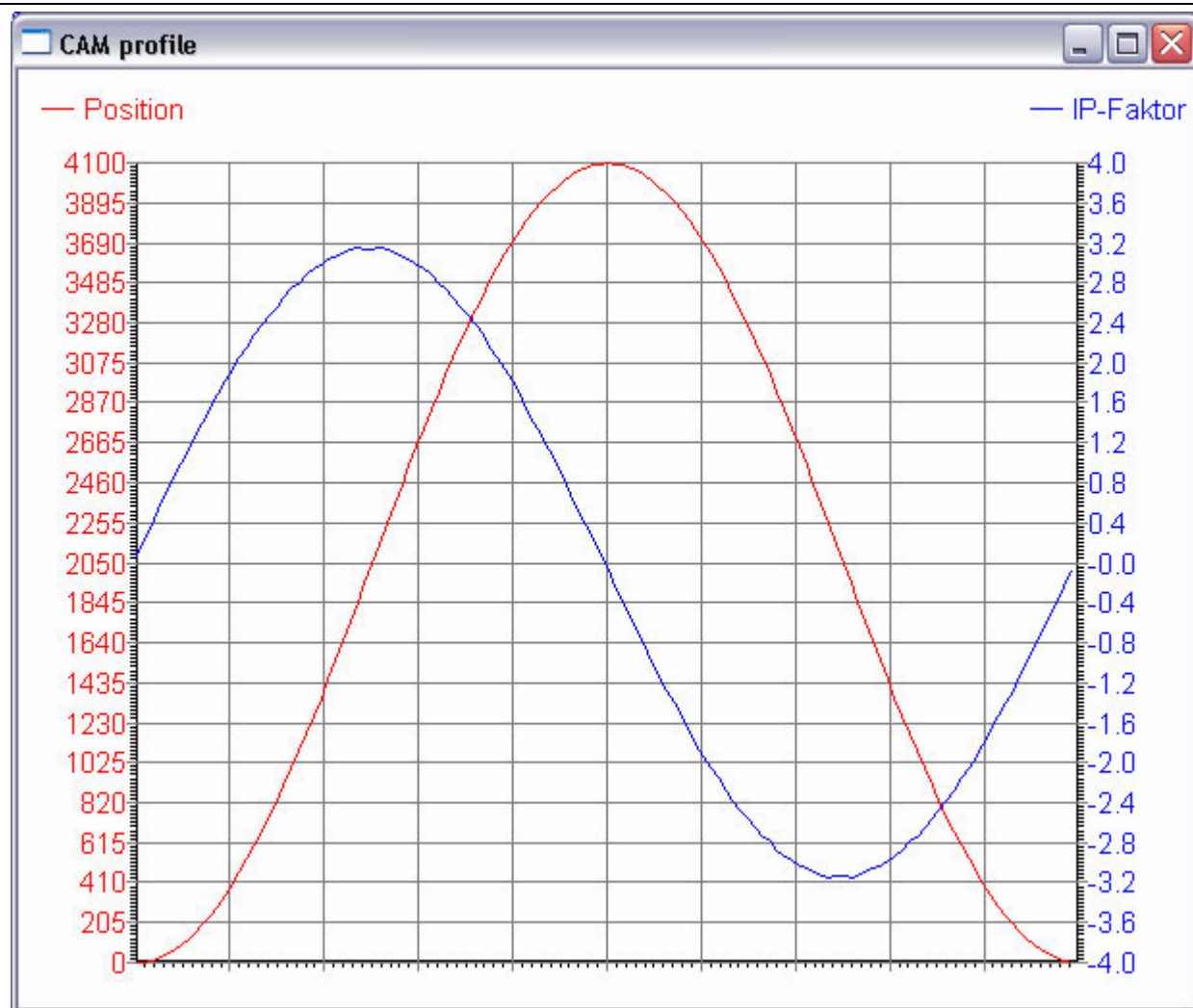
Incremental correction: ▾

Master cycle: incr

Slave cycle: incr

constant part: incr

Supporting point number:



Punch-profile

Cam profile ? X

Cam profile number:

Cam profile type:

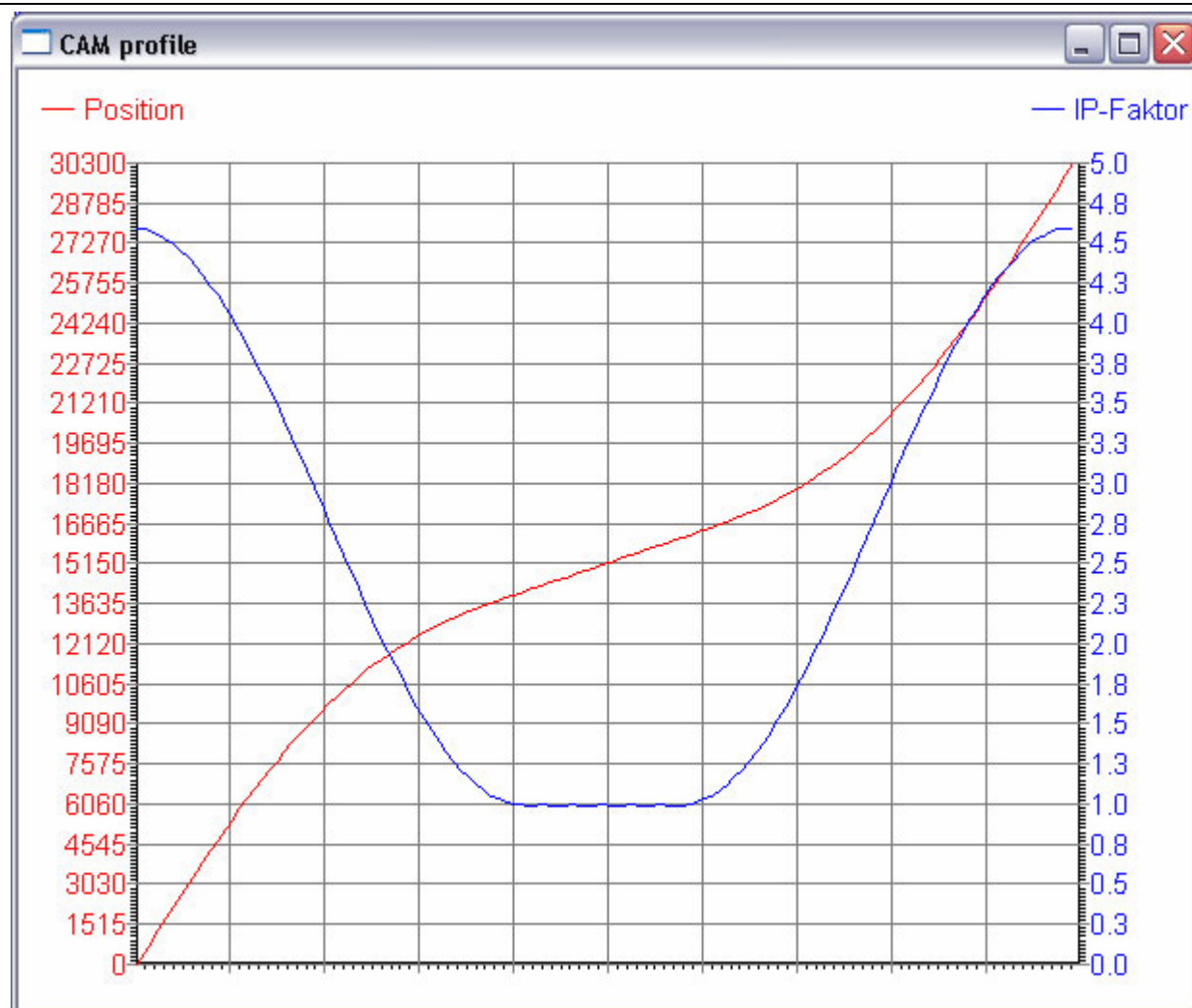
Incremental correction:

Master cycle: incr

Slave cycle: incr

synchronous part: incr

Supporting point number:



Explanation for the equations of the special profile type “Punch“:

This profile shape is used in cutting applications, where the knife is rotating in the direction of the constant material flow, to adjust different cutlengths. During the cut process the knife has to move synchron to the material for providing damages. When the knife has lost the contact to the material it can speed up or slow down to reach the required cutlength.

The most simple way to achieve the requirement is to put a straight line with the gradient of 1 between master and slave position.

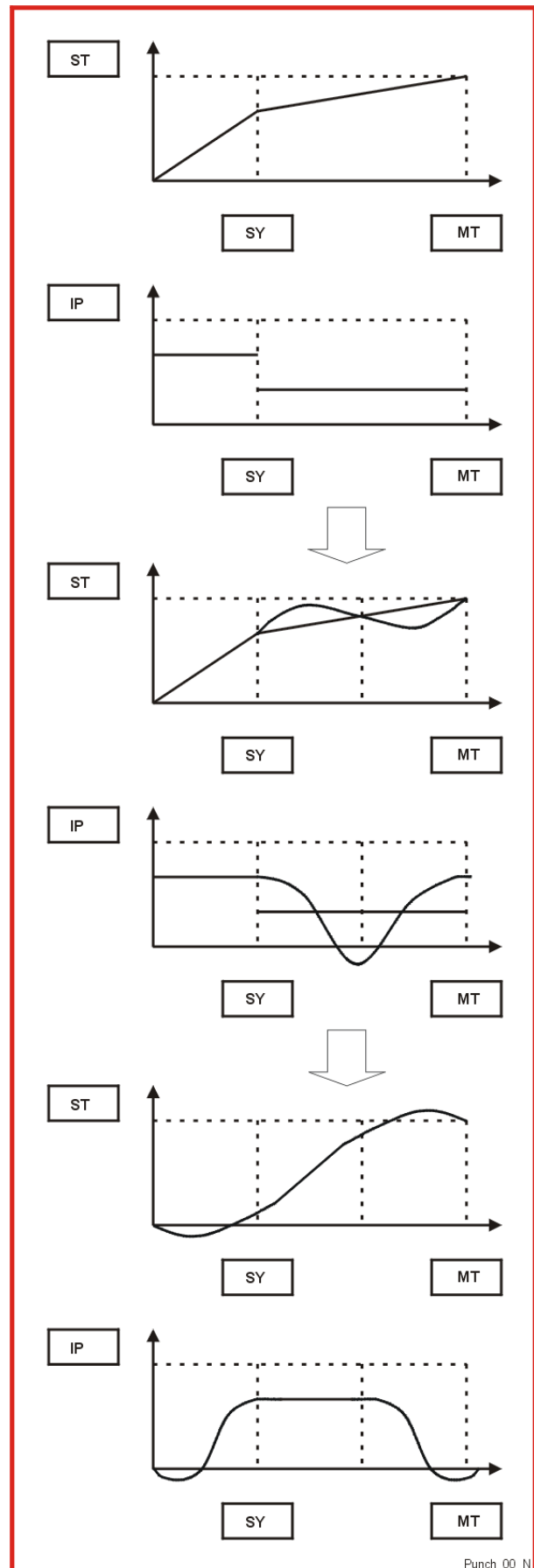
When the cut is finished (SY) another straight line can be put between (SY) and the last point of the profil.

This kind of profil will have a jump in the gearfactor between the two profile parts, that means high mechanical stress.

To provide this, a sinewave is overlapped to the movement in the asynchron region.

The amplitude of the sinewave can be calculated out of the continous condition in the point (SY).

For many applications it is necessary to have the cut in the middle of the cycle. So the asynchron part is divided at the end and in front of the synchronous part.



Linear-profile

Cam profile ? X

Cam profile number:

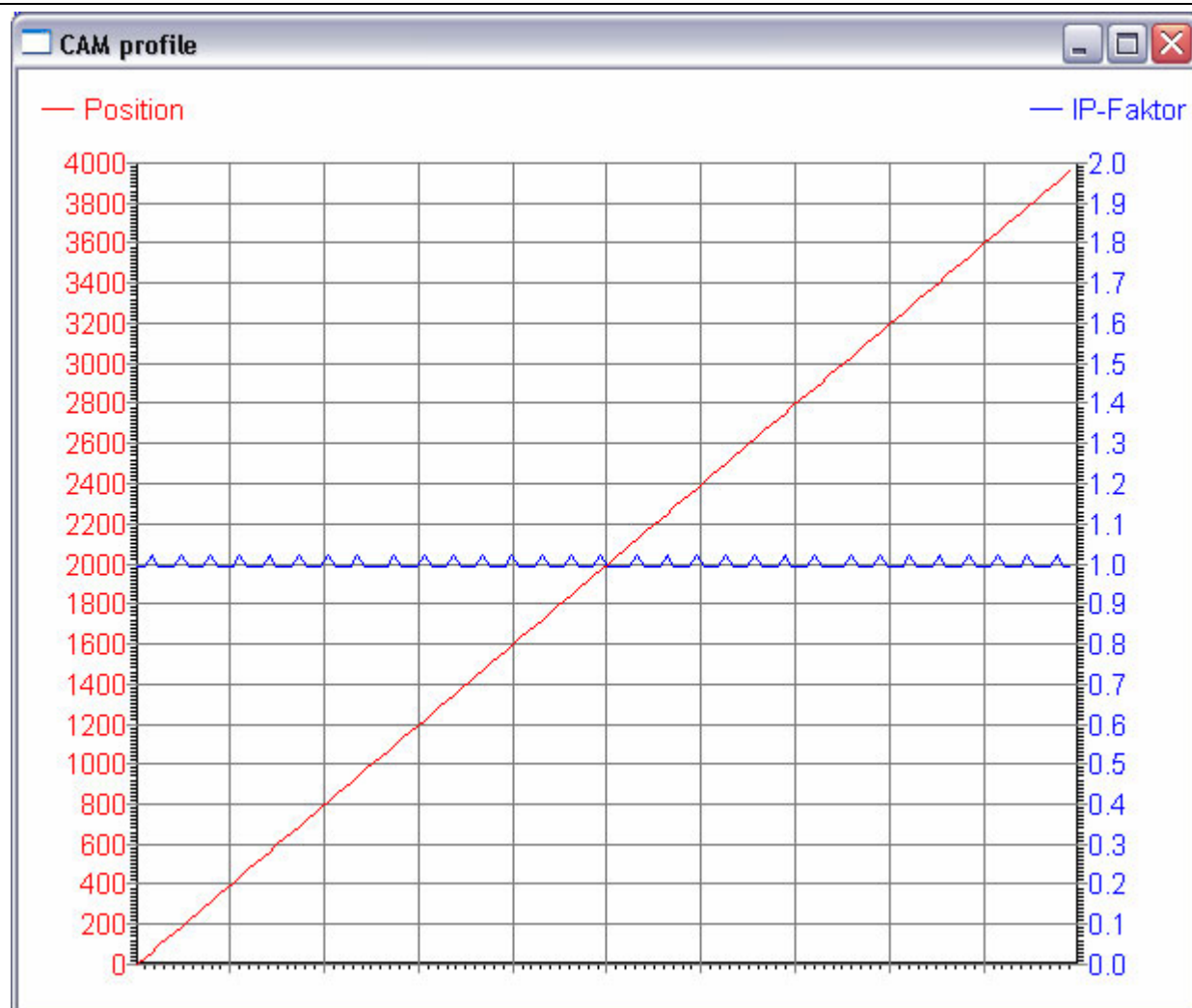
Cam profile type:

Incremental correction:

Master cycle: incr

Slave cycle: incr

Supporting point number:



Bilinear-profile

Cam profile ? X

Cam profile number:

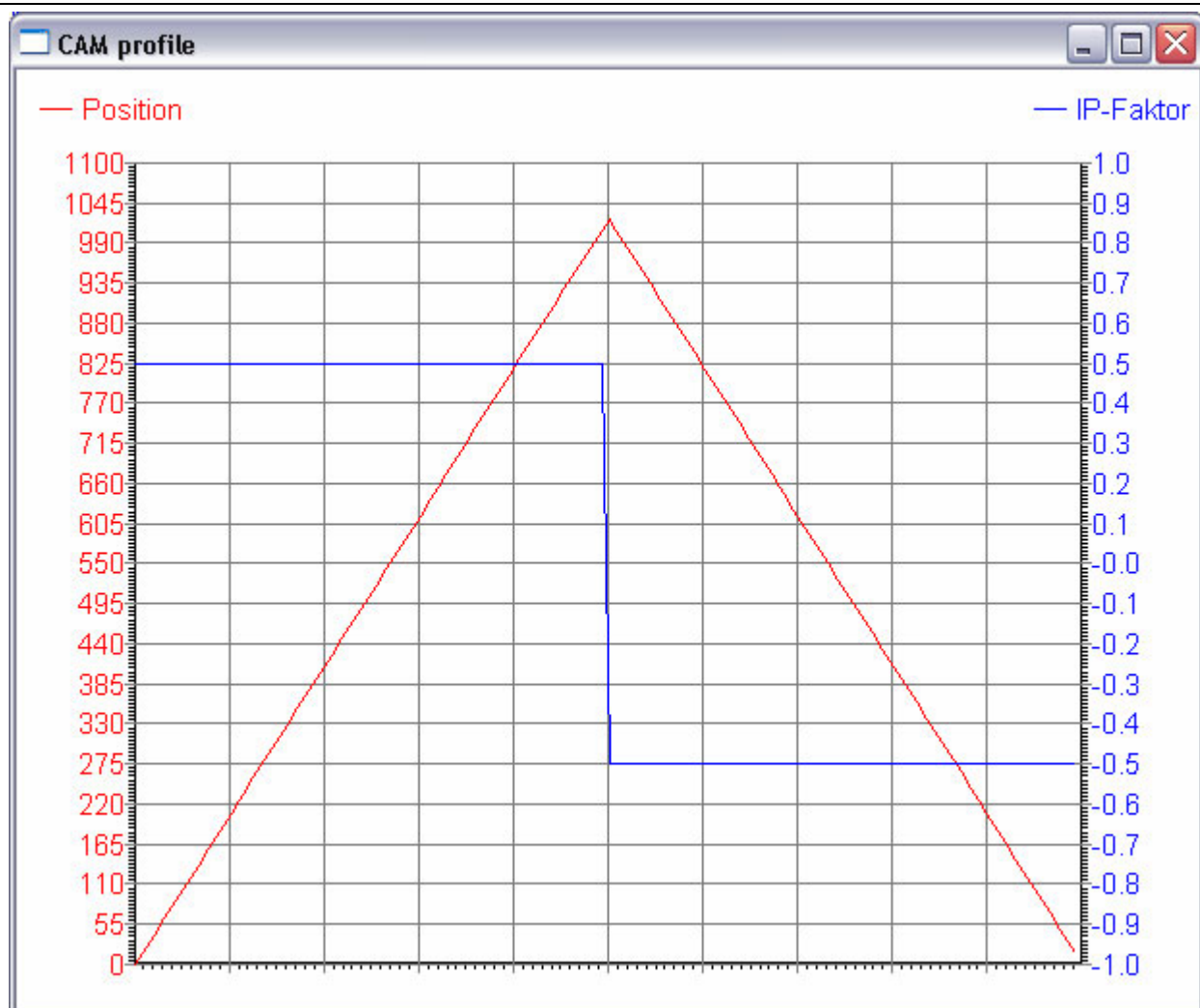
Cam profile type:

Incremental correction:

Master cycle: incr

Slave cycle: incr

Supporting point number:



The BIAS Interpreter allows with the extended Mathematic commands the access to the profilememory area.

With this method it is possible to (re-)calculate the CAM profiles direct in the drive.

The calculation can be done online and parallel to a running profile.

With the BIAS commands it is possible to switch over from the endposition of a profile to the startposition of a new profile.

The calculation of a Sine profile with 1024 supporting points and interpolationfactors takes appr. 3s.

For the accurate calculation of the profiles to BIAS interpreter allows to use up to 256 float- and double variables.

The following list shows the BIAS commands for the Profile memory access and the profile calculation.

BIAS command group „Mathematic commands“

Mathematic commands	No. permitted in programcycles...			
		BIAS	SPS	MATH
Profile commands				
[variable X] = profile value Y, Z	93	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
[X variable Y] = table [variable Z]	A2	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
profile initializing	91	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
profile cycle length	92	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
profile value Y, Z = [variable X]	94	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
table [variable X] = [Y variable Z]	A1	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
table [variable X]	A0	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
save table	97	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Floating point variable commands		BIAS	SPS	MATH
[D_variable X] = [D_variable Y] - [D_variable Z]	B1	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
[D_variable X] = [D_variable Y] * [D_variable Z]	B2	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
[D_variable X] = [D_variable Y] / [D_variable Z]	B3	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
[D_variable X] = [D_variable Y] + [D_variable Z]	B0	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
[D_variable X] = COS[D_variable Y]	B6	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
[D_variable X] = SIN[D_variable Y]	B5	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
[D_variable X] = SQRT[D_variable Y]	B7	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Variable type conversion				
[W variable X] = [Y variable Z]	A3	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

The access to the synchronous profiles is possible with the BUS-systems

Profibus
Interbus S
SucoNet K
CAN

and the serial link (RS232,RS22/485)

In the serial protocol for RS232 and RS422/485 it is possible to use the same commands as in the Profibus protocol.

You only have to take the frame for the serial communication: ESC, axisnumber, command 48 and the BCC character.

The communication protocol is defined in the bus manuals.

The following sheet shows the blocknumberdefinition for the synchronprofileblock and synchronprofile data access.

Profile 0 : 64 byte profildefinitionblock with blocknumber definition for bus access

mandatory

Blocknumber	databyte 0*	1	2	3
900h	char korregieren	char korrektmax	unsigned int profilepoints	
901h	unsigned int syncstartaddress		int deltamaster	
902h	int korrekturwert[0]		int korrekturwert[1]	
903h	int korrekturwert[2]		int korrekturwert[3]	
904h	int korrekturwert[4]		int korrekturwert[5]	
905h	int korrekturwert[6]		int korrekturwert[7]	
906h	int korrekturwert[8]		int korrekturwert[9]	
907h	long masterstroke			
908h	long slavestroke			
909h	long koppelfaktor (used from EASYRIDER profilecalculation)			
90Ah	long slavesyncwegink (used from EASYRIDER profilecalculation)			
90Bh	long slavesyncinkstart (used from EASYRIDER profilecalculation)			
90Ch	long slavesyncinkende (used from EASYRIDER profilecalculation)			
90Dh	char syncmode	reserved	reserved	reserved
90Eh	reserved	reserved	reserved	reserved
90Fh	reserved	reserved	reserved	reserved

...

Profile 1: 910h – 91Fh

.....

Profile 15: 9F0h –9FFh

Note !

*In the CAN BUS and Interbus blocktransfer protocol the block databytes 0 – 3 are databytes 2 –5 in the CAN Object!!!

*In the Profibus BUS blocktransfer protocol the block databytes 0 – 3 from the even blocknumbers are databytes 6 –9 and the block databytes 0 – 3 from the odd blocknumbers are databyte 10 –13 in the Profibus command!!!

Parameters of the synchrondatablocks

Slaveposition(i) { SP(i) }: Slaveposition of a profilpoint in increments. Datformat is long integer.

$$SP(i) = f(MP(i));$$

Interpolationfactor(i) {IP(i)}: gradient between two profilpoints*65536. Dataformat long integer.

$$IP(i-1) = 65536 * \{SP(i) - SP(i-1)\} / \{MP(i) - MP(i-1)\};$$

Condition:

$$\{MP(i) - MP(i-1)\} = \text{constant}!!!!$$

Blocknumber definition for bus access

Synchrondatablock 0 :

Blocknumber	databyte 0	1	2	3
1000h	slaveposition(i) SP(i)			
1001h	Interpolationfactor(i) {IP(i)}:			

Synchrondatablock 1

Blocknumber	databyte 0	1	2	3
1002h	slaveposition(i) SP(i)			
1003h	Interpolationfactor(i) {IP(i)}:			

.....

Synchrondatablock 2047

Blocknumber	databyte 0	1	2	3
1FFEh	slaveposition(i) SP(i)			
1FFFh	Interpolationfactor(i) {IP(i)}:			

Note !

*In the CAN BUS and Interbus blocktransfer protocol the block databytes 0 – 3 are databytes 2 –5 in the CAN Object!!!

*In the Profibus BUS blocktransfer protocol the block databytes 0 – 3 from the even blocknumbers are databytes 6 –9 and the block databytes 0 – 3 from the odd blocknumbers are databyte 10 –13 in the Profibus command!!!

Definition of EASYRIDER file format for CAM-profile im- and export

The EASYRIDER is able to im- and or export cam profiles up from the version 5.11B.

The 1st step is to create an import file in a special file format.

The used file extension is“.prf“

The file is a textfile, all parameters are separated by tabs.

There are a mandatory and a optional parameter part.

mandatory

Example for CAM-profile file:

Profiletype ¹⁾	2									
Masterstroke	1263									
Slavestroke	25666									
Incrementalcorrection ²⁾	1									
Correctionsteps	4									
Correctionvalues	3	5	7	20	-1	-1	-1	-1	-1	-1
Profilepoints	128									
Syncstartaddress ³⁾	128									

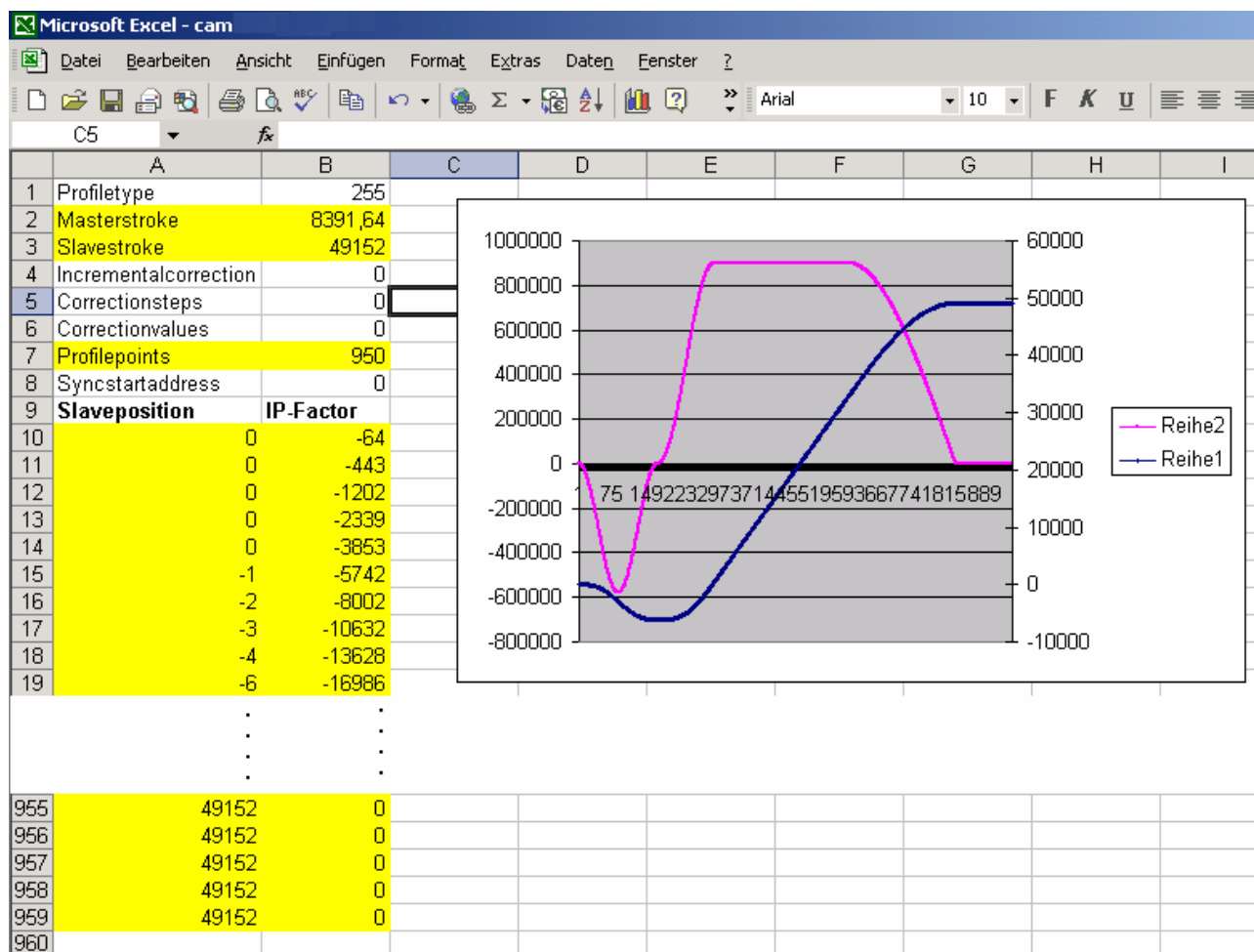
Slaveposition	IP-Factor
0	112563
15	337417
62	561459
3759	3321893
.	.
.	.
.	.
.	.
6236	3990814
6784	4096938
7346	4193192
7922	4279343
8510	4355186

¹⁾ default value 255 (1-20 are reserved for EASYRIDER-profiletypes)

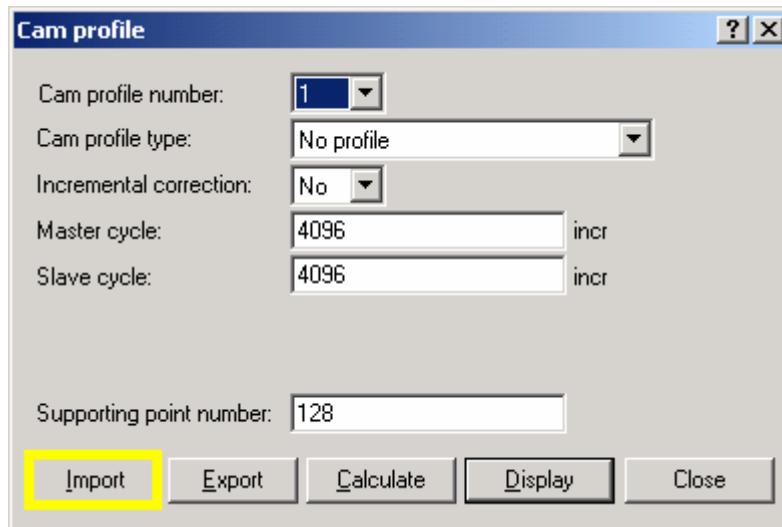
²⁾ 0 = no incremental corrections executed, values for correction steps and correction values ignored (default value 0)

³⁾ if defined and not equal -1: no verifications are executed

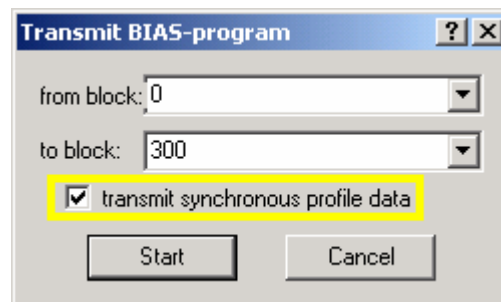
A comfortable editor is the EXCEL program.
 The restrictions for the expected fileformat is automatically adjusted.
 Just save the file with the extension *.prf.



The 2.nd step is the import of this file with the EASYRIDER software.



The 3.rd step is to integrate the Profile in a BIAS program and download the BIAS program with the synchronous profile data to the drive.



630 Series Type	631	635	637 /3G	637+ /4G	637f /5G
Sample time [μs] Current Loop	211	211	211	105	105
P_GAIN Current Loop	0,245* P_Gain* Ucc/ I _{NR} * V/A	0,245* P_Gain* Ucc/ I _{NR} * V/A	0,245* P_Gain* Ucc/ I _{NR} * V/A	0,245* P_Gain* Ucc/ I _{NR} * V/A	0,245* P_Gain* Ucc/ I _{NR} * V/A
Integration Time const. Current Loop [ms]	I_Gain*3	I_Gain*3	I_Gain*3	I_Gain	I_Gain
Sample time [μs] Speed Loop	211	211	211	105	105
P_GAIN Speed Loop	3,34e-4* P_Gain* I _{NR} *A/rpm	3,34e-4* P_Gain* I _{NR} *A/rpm	3,34e-4* P_Gain* I _{NR} *A/rpm	3,34e-4* P_Gain* I _{NR} *A/rpm	3,34e-4* P_Gain* I _{NR} *A/rpm
Integration Time const. Speed Loop [ms]	I_Gain	I_Gain	I_Gain	I_Gain	I_Gain
Sample time [μs] Speed-Filter/ Number of samples	211/6... 211/32	211/6... 211/32	211/6... 211/32	106/4	106/4
Sample time [μs] Position Loop	1890	1890	1890	844	105
P_GAIN Position Loop	P_Gain* 0.015* rpm/Inc	P_Gain* 0.045* rpm/Inc	P_Gain* 0.045* rpm/Inc	P_Gain* 0.015* rpm/Inc	P_Gain* 0.015* rpm/Inc
Integration Time const. Position Loop [ms]	1800/ I_Gain	1800/ I_Gain	1800/ I_Gain	844/ I_Gain	844/ I_Gain
V_Gain Position Loop	100%= Trajectory speed	100%= Trajectory speed	100%= Trajectory speed	100%= Trajectory speed	100%= Trajectory speed
Sample time [μs] ramp- Filter/ Number of samples	1890/ 0-255	1890/ 0-255	1890/ 0-255	844/ 0-255	844/ 0-255
Resolution Positionfeedback	4096Inc/rev for max speed =+/- 12000rpm 16384Inc/rev for max speed =+/- 4000rpm 1)	4096Inc/rev for max speed =+/- 12000rpm 16384Inc/rev for max speed =+/- 4000rpm	4096Inc/rev for max speed =+/- 12000rpm 16384Inc/rev for max speed =+/- 4000rpm	65536 Inc/rev	65536 Inc/rev =16 bit Resolver, HIPERFACE® 1024 Inc/encoder_period ,SIN/COS Example: 1024 encoder_periods/rev 1048576 Incr/rev =20bit
Resolution Speedfeedback	1,44 rpm on high resolution 5,76 rpm on low resolution 1)	1,44 rpm on high resolution 5,76 rpm on low resolution	1,44 rpm on high resolution 5,76 rpm on low resolution	1,08rpm	1,08rpm Resolver, HIPERFACE 69,75rpm/encoder_periods SIN/COS Example:1024 encoder_periods/rev 0,068 rpm

630 Series Type	631	635	637 /3G	637+ /4G	637f /5G
Resolution speed setpoint	0,5 rpm	1,44 rpm on high resolution 5,76 rpm on low resolution	1,44 rpm on high resolution 5,76 rpm on low resolution	0,54 rpm	0,54 rpm W_V_SHIFT=0 0,54/2 rpm W_V_SHIFT=1 0,54/4 rpm W_V_SHIFT=2 0,54/8 rpm W_V_SHIFT=3 0,54/16rpm W_V_SHIFT=4
Maximum Speed	+/-4000rpm high resolution +/-12000rpm low resolution	+/-4000rpm high resolution +/-12000rpm low resolution	+/-4000rpm high resolution +/-12000rpm low resolution	+/-16000rpm	+/-16000rpm +/-8000rpm +/-4000rpm +/-2000rpm +/-1000rpm
PWM frequency	4.7KHz 2)	9.4KHz	4.7KHz	4.7KHz	4.7KHz
PWM-Resolution	9 bit /1step =0,9Veff at 310V Ucc	10 bit /1step =0,45Veff at 310V Ucc	9 bit /1step =0,9Veff at 310V Ucc	2623 steps/ 1step=0,175Veff at 310V Ucc	4196 steps/ 1step=0,109Veff at 310V Ucc
Resolution Current Messuerment	512Digits =2,7*I _{NR}	512Digits =2,7*I _{NR}	512Digits =2,7*I _{NR}	512Digits =2,7*I _{NR}	512Digits =2,7*I _{NR}
Resolution Analog-inputs	12bit A_In1	12bit A_In1 10bit A_In2	12bit A_In1 10bit A_In2	14bit A_In1 10bit A_In2	14bit A_In1 10bit A_In2
Resolution/ internal resistance / Timeconstant of Analog-outputs 3)	-----	7 bit 10 kOhm 47 µs	7 bit 10 kOhm 47 µs	X10.17: 8bit X10.6: 10bit 1.8 kOhm 8.5 µs	X10.17: 8bit X10.6: 10bit 1.8 kOhm 8.5 µs
Resolution/ Max. frequency X40 Input	200 KHz	200 KHz	200 KHz	200 KHz	200 KHz
Resolution Max. frequency/ Deadtime 4) X40 Output	64/128/256/ 512/1024/ 2048/4096/ up to 125 µs	64/128/256/ 512/1024/ 2048/4096/ up to 125 µs	64/128/256/ 512/1024/ 2048/4096/ up to 125 µs	64/128/256/ 512/1024/ 2048/4096/ up to 45 µs	64/128/256/ 512/1024/ 2048/4096/ up to 45 µs
Time delay Latchinputs	47 µs RC-filter +10..20 µs softw.	47 µs RC-filter +10..20 µs softw.	47 µs RC-filter +10..20 µs softw.	47 µs RC-filter +10..20 µs softw.	47 µs RC-filter +10..20 µs softw.

- 1) The low position resolution is implemented on the 631 in that way that the actual position is incremented or decremented by 4 for one counter step. This keeps the position values and control gains compatible to the high resolution parameters
- 2) From Firmware 6.14a it is possible to double the PWM Frequency
- 3) The Analog outputs were implemented as PWM and RC-Filter
- 4) The encodersimulation sends every 644 µs on the 631 /635 / 637 and every 422 µs on the 637+ a pulse package. There is always a deadtime between two packages.

Modification Record

Version	Modification	Chapter	Date	Name	Comment
V0102	New		11.12.2002	T.Saladin R.Krebs N.Dreilich	
V0203	Documentation - layout		21.01.2003	N.Dreilich	